3-2007

On-Demand Key Distribution for Mobile Ad-Hoc Networks

Daniel F. Graham

ON-DEMAND KEY DISTRIBUTION FOR MOBILE AD-HOC NETWORKS

THESIS

Mr. Daniel F. Graham

AFIT/GCS/ENG/07-12

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

AFIT/GCS/ENG/07-12

ON-DEMAND KEY DISTRIBUTION FOR MOBILE AD-HOC NETWORKS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Science

Mr. Daniel F. Graham, BS

March 2007

AFIT/GCS/ENG/07-12

ON-DEMAND KEY DISTRIBUTION FOR MOBILE AD-HOC NETWORKS

Mr. Daniel F. Graham, BS

Approved:

/SIGNED/	5 Mar 2006
Dr. Kenneth M. Hopkinson (Chairman)	Date
/SIGNED/	5 Mar 2006
Dr. Rusty O. Baldwin (Member)	Date
/SIGNED/	5 Mar 2006
Maj Scott R. Graham, Ph.D (Member)	Date

Abstract

Mobile ad-hoc networks offer dynamic portable communication with little or no infrastructure. While this has many benefits, there are additional shortcomings specific to wireless communication that must be addressed. This research proposes gossip-based on-demand key distribution as a means to provide data encryption for mobile ad-hoc networks.

This technique uses message keys to avoid encrypting and decrypting a message at every node. Other optimizations used include secure channel caching and joint rekey messages. The use of gossip makes the scheme robust to node failure.

Experimental results show only a 15% increase in end-to-end delay with a node failure rate of 10%. The percentage of messages successfully delivered to nodes stays between 91-98% under the same 10% node failure rate. The network load is distributed to all nodes in the group preventing overload and single points of failure.

To Jill, this endeavor would not have been possible without you.

Acknowledgments

I would like to thank God for blessing me with this opportunity to grow in more ways than just the knowledge gained.

I would also like to express my sincere appreciation to my faculty advisor, Dr. Kenneth Hopkinson, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated. I would like to thank my sponsor, Dr. David Luginbuhl, from the Air Force Office of Scientific Research for both the support and latitude provided to me.

Lastly, I would like to thank my friends and family for their encouragement over the course of this work.

Daniel F. Graham

Table of Contents

Page

Abstract iv
Acknowledgments vi
Table of Contents vii
List of Figures x
List of Tables xii
I. Introduction 1
1.1 Background 1
1.2 Research Objectives 2
1.3 Overview 3
II. Literature Review 4
2.1 Chapter Overview 4
2.2 Background 4
2.2.1 Unicast Security 4
2.2.1.1 Public Key 5
2.2.1.2 Symmetric Key 5
2.2.2 Group Communication 6
2.2.2.1 Multicast 7
2.2.2.2 Mobile Ad-hoc Networks 9
2.2.3 Secure Group Communication 10
2.2.4 Required Trust 10
2.3 Current Research 11
2.3.1 Iolus 11

2.3.2 Cipher Sequences 13
2.3.3 Fast Authenticated Key Establishment....14
2.3.4 Antigone 14
2.3.5 Secure Channel Caching....15
2.3.6 Gossip using Catalog Messages 16
2.3.7 Gossip as a Self-Sufficient Protocol....16
2.4 Summary....17
III. Methodology 18
3.1 Problem Definition 18
3.1.1 Goals and Hypothesis 19
3.2 Approach 19
3.2.1 Secure Unicast Links 19
3.2.2 Minimize Encryption....20
3.2.3 Membership....23
3.2.4 Caching Secure Unicast Channels....24
3.3 System Boundaries 25
3.4 System Services....25
3.5 Workload 26
3.6 Performance Metrics 26
3.7 System Parameters....26
3.8 Workload Parameters 27
3.9 Factors 27

3.10 Evaluation Technique 28
3.11 Validation 28
3.12 Experimental Design 29
3.13 Summary 29
IV. Results and Analysis 30
4.1 Chapter Overview 30
4.2 End-to-end Delay 30
4.3 Network Load 38
4.4 Percent of Messages Delivered 42
4.5 Summary 44
V. Conclusion 45
5.1 Chapter Introduction 45
5.2 Conclusion 45
5.3 Recommendations for Future Research 46
Appendix A: Additional Charts 47
Bibliography 58

List of Figures

Page

Figure 1. Unicast distribution. 7

Figure 2. Multicast tree distribution..... 8

Figure 3. Iolus-based network..... 12

Figure 4. Two Cipher Sequences 13

Figure 5. a. Antigone Joint Message b. Full Antigone Rekey Message 15

Figure 6. A single gossip 16

Figure 7. A single message being gossiped without message keys 20

Figure 8. Send procedure 21

Figure 9. Receive procedure 21

Figure 10. Message Delivery 21

Figure 11. Receiving a message key..... 22

Figure 12. Receiving a key request..... 22

Figure 13. A single message being gossiped with message keys 23

Figure 14. 51 node mesh topology..... 27

Figure 15. Average End-to-End Delay for 51 node scenarios 31

Figure 16. Average End-to-End Delay for 101 node scenarios 32

Figure 17. Average End-to-End Delay for 2 group scenarios 32

Figure 18. End-to-End delay standard deviation 34

Figure 19. 51 Node Scenarios Maximum End-to-End delay 34

Figure 20. 101 Node Scenarios Maximum End-to-End delay 35

Figure 21. 2 Group Scenarios Maximum End-to-End delay 35

Figure 22. Average key distribution End-to-End delay 36

Figure 23. Average message key distribution End-to-End delay for 10% node fault 37

Figure 24. Maximum message key End-to-End delays for 10% node fault 37

Figure 25. Average message send rate 39

Figure 26. Maximum message send rate 39

Figure 27. Average send rate per node 40

Figure 28. Maximum send rate per node in 51 node scenarios 40

Figure 29. Maximum send rate per node in 101 node scenarios 41

Figure 30. Maximum send rate per node in 2 group scenarios 41

Figure 31. Percentage of gossip messages received in 51 node scenarios 42

Figure 32. Percentage of gossip messages received in 101 node scenarios 43

Figure 33. Percentage of gossip messages received in 2 group scenarios 43

List of Tables

Page

Table 1. Factors Settings 28

Table 2. Node Failure Rate and End-to-End percent increase for all scenarios 33

ON-DEMAND KEY DISTRIBUTION FOR MOBILE AD-HOC NETWORKS

I. Introduction

1.1 Background

Mobile ad-hoc networks (MANETs) have the potential to provide communication to a large group of nodes while decreasing the amount of infrastructure needed to establish and maintain communication. Certain applications have communication requirements that can only be satisfied by wireless networking. Wireless networking is used everyday when talking on a cell phone or surfing the Internet on a laptop at a coffee shop. These common applications often have some sort of infrastructure. This may be a wireless access point in the coffee shop or the thousands of cell phone towers on top of buildings, beside highways, or upon hills.

However, in many cases, especially in military applications, it is not feasible to rely so heavily on an established network infrastructure. When entering hostile territory, the infrastructure may not exist or the existing infrastructure may not be available. Furthermore, establishing an infrastructure can be costly and will likely become a target for the enemy. MANETs enable communication wherever nodes are present avoiding the costs and limitations of a fixed infrastructure.

Much of the current research and networking protocols assume a wired network. Moving from a wired to a wireless environment introduces considerations that are not a concern in wired networks. One such consideration is the likelihood nodes will

experience intermittent connections. The connection problem could be isolated to a single link, like when an obstacle blocks transmission. The problem could also affect all links for a particular area, perhaps due to interference. Reliable group communication protocols attempt to send all updates to intermittent nodes. Sending to a node during its down time can increase the load on the network.

Gossip-based protocols communicate using epidemic communication patterns. Messages are distributed to randomly selected nodes and probabilistically delivered to all nodes depending on each nodes connectedness to the network, known as their health. A healthy node will likely receive the desired amount of updates, while an unhealthy node will likely receive only as many updates as its intermittent links will allow. Gossip-based networking attempts to overcome the unreliability of wireless networks.

Many applications require secure networks in which nodes transmit data in encrypted form. This is particularly essential for wireless communication as all transmissions are subject to monitoring. Implementing security is challenging when using group communication, and is further complicated when operating over wireless networks. A key distribution scheme for such environments must ensure all nodes to receive keys in a timely, efficient manner. The scheme must also be able to support any unhealthy nodes in the network.

1.2 Research Objectives

This research develops a key distribution scheme that adds data encryption to a gossip-based protocol. The scheme adds little overhead to the network and delivers keys

to the extent that the health of nodes allows. This thesis demonstrates a security protocol and compares it to a secure reliable Iolus-based distribution scheme.

1.3 Overview

Chapter 2 provides background information on network encryption protocols and group communication. Chapter 3 states the problem, introduces a method for overlaying security on a gossip-based protocol, and discusses the experiments to analyze this approach. Chapter 4 presents the data resulting from the experiments. Finally, Chapter 5 presents conclusions and proposes areas of further research to improve the protocol.

II. Literature Review

2.1 Chapter Overview

This chapter reviews topics in gossip-based key distribution protocol including security for unicast communication and how it relates to group communication. Multicast communication is also discussed and current research on secure group network protocols considered.

2.2 Background

2.2.1 Unicast Security

Unicast communication is network communication in its most fundamental form, one node communicating with another node. Transmitted data is considered secure if no node other than the receiver is able to interpret the data. Securing unicast communication can be accomplished in various ways. For instance, a dedicated line can be used for secure transmission assuming the line is not compromised through some form of wiretapping. While this has inherent benefits, the use of a dedicated line is rarely feasible as a cost-effective solution. More importantly, dedicated lines are impossible if nodes are mobile. Therefore, nearly all network communication is done over shared access media. Whether the communication is wired or wireless, other nodes will receive the data being transmitted.

Several security protocols allow nodes to use a shared network securely. Nearly all network security protocols fall into to two major categories, public key or symmetric key. Public key algorithms use separate keys for encryption and decryption. Symmetric

key algorithms use a single key for encryption and decryption and are typically much faster than public key algorithms.

2.2.1.1 Public Key

The Rivest, Shamir, and Adleman (RSA) public key encryption algorithm [Rivest78] exploits the complexity of factoring a product of large primes to make deciphering encrypted data very difficult. Two keys are used in the transmission of data. One key is made public and is used by other nodes to encrypt data that will be sent to the node to which the private key belongs. The other key is kept private and is used solely by the receiving node. The node decrypts the encrypted data sent to it using the private key. Due to the complexity of the math involved in the encryption and decryption processes, the RSA algorithm, like many public key encryption algorithms, tends to be slow compared to symmetric key algorithms.

Using public key algorithms, nodes can send encrypted data to a receiving node as well. Each sending node uses the same public key to encrypt the data being sent to a single receiving node. Confidentiality is maintained since the only node capable of decrypting the data is the node that has the private key.

2.2.1.2 Symmetric Key

Symmetric key algorithms, such as Rivest Cipher 5 (RC5) [Rivest95], are often much less computationally intensive than public key encryption algorithms making symmetric key algorithms faster and easier to implement than public key algorithms. The principle shortcoming is the algorithms require a secret key be shared by the nodes prior

to encrypting data. Therefore, a secret channel between the nodes must be used to exchange the shared key.

Sharing a symmetric key carries a much greater significance than sharing a public key. A node receiving a symmetric key, in addition to encrypting data, will be able to decrypt everything encrypted with that symmetric key. Once a key has been shared with a node, that node has access to all data encrypted with that key; future, present, and past. For two nodes, this means very little. A key is generated and shared at the beginning of the session. When the session is done, the sender drops the shared key. Confidentiality problems occur if a symmetric key is reused in sessions with other nodes. Further considerations must be made when three or more nodes are communicating using a single symmetric key as will be discussed later.

2.2.2 Group Communication

Group communication is network communication between a specific set of nodes. There is a growing need to distribute data to large groups of nodes. From streaming media to reconnaissance video, there are applications in both the commercial and defense arenas. Group communication in a unicast manner is inefficient; a single sending node must transmit the entire message separately to each receiving node in the group, as in Figure 1. This causes unicast distribution to scale poorly.

Effective group communication protocols must be able to transmit large amounts of data with a relatively small amount of overhead. For example, a GPS position for a location of importance could be distributed to large group of nodes by simply setting up a unicast link to each node without regard to its inefficiency since the amount of data is

small. However, a video stream will overwhelm a node trying to unicast the stream to even a small group because the sender must individually encrypt and send the data to each receiving node.

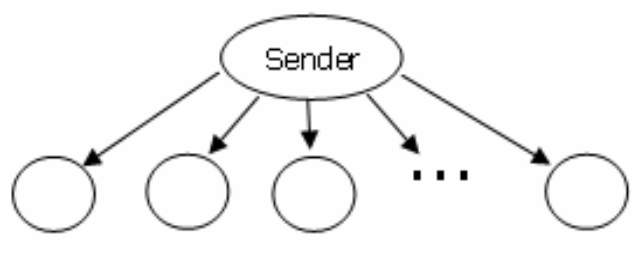


Figure 1. Unicast distribution.

Effective group communication must also be able to handle large groups. Each additional receiver in a group is potentially another copy of the message that must be encrypted and sent. Average delay to send a message grows exponentially as the amount of data approaches network capacity. Thus, while it is feasible to unicast distribute small amounts of information to a small group, but problems arise and are magnified with bigger groups and larger data sizes.

Another factor is an expectation of timeliness. Data sent using group communication is assumed to be useful when received. For example, the usefulness of threat position data is nil if the threat has already been encountered. For group communication to be effective it must be timely, able to handle large amounts of data, and able to scale to large groups.

2.2.2.1 Multicast

A multicast communication protocol is addresses some of the scalability issues in group communication. Multicast differs from broadcast in that multicast messages are

intended for a specified group of nodes. Ideally, a node sends a multicast message once and it is reliably received by all of the other nodes in the group with no duplicate packets being transmitted.

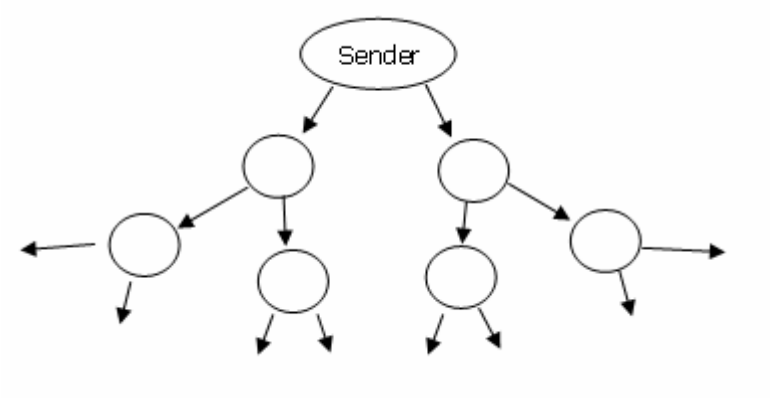


Figure 2. Multicast tree distribution

A constant load send method in group communication forms a tree between a sender and receivers, as in Figure 2. When a send request is made, the receiver of the message forwards it to each of the children in the multicast tree. The children that receive the message then forward the message to their children which continues until all nodes receive the message.

The multicast tree has two major drawbacks. First, if a node fails, all child nodes will no longer receive messages. Second, some knowledge of the topology of the nodes must be known to establish the multicast tree. Otherwise, the result may be more of a logical overlay than it is a multicast tree which may not be timely or efficient. Finally, this method lacks mechanisms to ensure that it is robust or scalable.

2.2.2.2 Mobile Ad-hoc Networks

Networked devices are expanding beyond the static world of traditional wired networking. Wireless network technology has become widely available, and it allows devices to communicate with one another untethered by wires. Furthermore, devices not only stay connected to each other, but have the ability to move and discover other devices. These dynamic networks are known as mobile ad-hoc networks (MANETs).

Along with the benefits of MANETs, there are complications that must be properly addressed. One such complication is the reliability of the wireless connections. Data transmitted wirelessly is typically sent via radio frequencies (RF). Examples include 802.11 and Bluetooth. RF transmission is prone to many types of failures that are not as common in wired transmissions.

Another shortcoming of MANETs is the mobility of the nodes in the network. A node that was previously connected to a group of nodes might encounter an obstacle that prevents communication. The break in communication may be brief or extended and may cause the node to become isolated. Sometimes the node may still be connected but via different links.

The unreliability of wireless links in addition to problems caused by the roaming nodes leads to a high probability that nodes will disconnect from the network and connect at a later time. Roaming nodes may also join and leave various connected components of the network over time as they move. The rate of join and leave events in the network is known as the churn rate. A network with a high churn rate has difficulty in maintaining

reliability because in attempting to achieve reliability in an unreliable environment, it increases the overhead of the network.

2.2.3 Secure Group Communication

A group must have a key distributed to all of its member nodes to establish secure communication. To satisfy forward and backward confidentiality, the group key must change on each join and leave. This can be accomplished on a join by using the previous group key to distribute the new group key and using secure unicast to send the new key to the joining node. Processing a leave is more difficult as the previous key cannot be used since the leaving member can retain the previous key and decrypt further communication using the key. A typical solution sends the new group key to each remaining node independently using multiple unicast secure transmissions. Mittra describes these issues as "1 affects n" failures on joins and both "1 affects n" and "1 does not equal n" failures on leaves [Mittra]. "1 affects n" is a failure where a single event requires an action to be taken for all other members in a group. A "1 does not equal n" failure is a situation where a group cannot be acted on as a group and each member must be considered separately.

2.2.4 Required Trust

With no restrictions on topology, any node in a group has the ability to covertly establish a subgroup not authorized by the group. The node that is a member of the group can distribute all decrypted messages it receives to its covert subgroup. Mobile ad-hoc

networks make this even easier since a covert channel can be created simply by changing frequency.

Due to this inherent vulnerability, the entire membership of a group must be trusted to maintain the membership of the group appropriately. This means that all nodes must be informed of the group membership so they can make the appropriate choices as to send or not send a message or key to another node.

2.3 Current Research

2.3.1 Iolus

The Iolus protocol [Mittra] is a method for alleviating "1 affects n" and "1 does not equal n" failures by breaking multicast groups into smaller subgroups and organizing the subgroups into a hierarchical distribution tree. Each subgroup has a group security agent (GSA) that maintains keying material specific to that subgroup. The subgroups are joined together by the GSAs that are members of a subgroup higher in the tree in addition to their own subgroup. They are responsible for providing communication between the two groups as well as providing communications for the membership of the subgroup they are serving. Figure 3 shows an Iolus group configuration.

To bridge a subgroup, message encrypted using one subgroup's key is reencrypted using to the key of the other group. To avoid decrypting every message to plain text and encrypting the plain text for the other subgroup, a message key can be used to encrypt the message so only the message key must be reencrypted. The GSA is responsible for processing all joins and leaves in the subgroup.

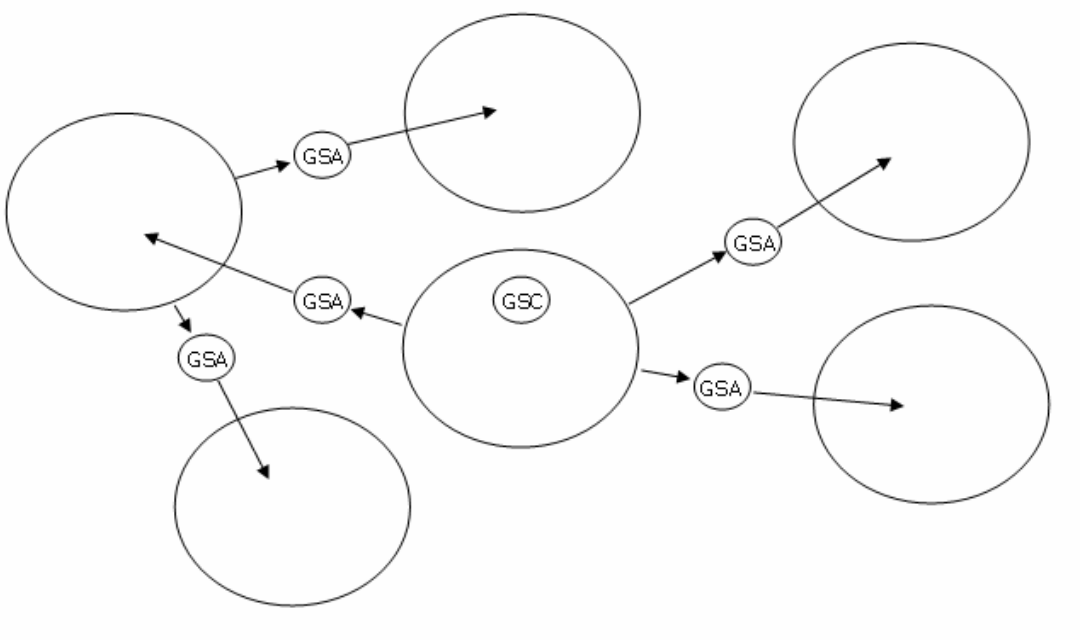


Figure 3. Iolus-based network.

The proposed method to process a join is to multicast the new key using the current group key to the current membership. The GSA must then establish a unicast secure channel to the joining node and send the new key through the established channel. This channel is kept open for the duration of the nodes membership in the subgroup. Thus, the GSA must keep a list of all of the open secure channels to each of the subgroup members.

A leave is more expensive as the current key cannot be used. The leave process creates a new key and sends it to each of the members in the group through the secure unicast channel. An alternative approach encrypts the new subgroup key using the key for each secure channel and merges the result into one large message, which is multicast to the subgroup.

Iolus reduces the effect of failures caused by joins and leaves, but relies on a single intermediary to bridge subgroups. This does not suit mobile ad-hoc networks well since the failure of a single node or an intermittent link can partition the network. Iolus also does not appear to take advantage of overlapping groups. Each group must be maintained by a new set of GSAs. At the very least, GSAs must keep separate keying material and access control list (ACL) for every group.

2.3.2 Cipher Sequences

Cipher sequences address some of the issues of Iolus [Molva]. The group is divided into sub groups similar to Iolus. However, instead of being connected to other subgroups via an intermediary, sub groups are connected to the root node through a series of intermediate nodes, called inner nodes. Each of the inner nodes is given part of a cipher sequence. As a message is sent through the inner nodes, each node applies its piece of the cipher to the message. Figure 4 shows two cipher sequences in one map.

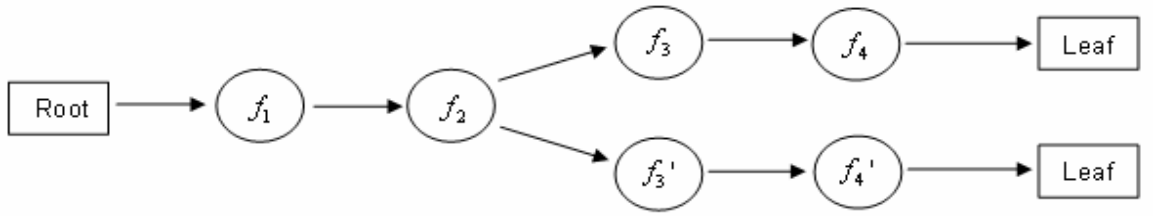


Figure 4. Two Cipher Sequences

A benefit of this approach is that the intermediaries are prevented from handling plain-text data in the manner that GSAs must do in Iolus. This is important because, in contrast to Iolus, inner nodes are not members of the group. The inner nodes' only function is to encode the cipher sequences represented by the "f" characters in Figure 4.

A related benefit of cipher sequences is that messages are encrypted separately for each group. Each leaf is connected to the root through a unique set of inner nodes so the resulting encryption of the data by the inner nodes is unique. This isolates the effect of any compromised node to a single leaf. The compromise of one leaf does not compromise the remaining leaves.

2.3.3 Fast Authenticated Key Establishment

Huang et al. proposes a method for nodes to efficiently self-authenticate and share a key [Huang] by setting up each node offline prior to deployment. A certificate is generated with the help of a trusted computation server. The certificate is just a public key, device ID, and expiration date that has been signed by the certificate authority. Elliptical Curve Cryptography is used to take advantage of its small key sizes and low communication complexity. The protocol is designed for distributed sensor networks and is intended to lower the communication and computational load on each sensor since sensor nodes have limited computational ability and are often power constrained. The protocol uses certificates when the nodes are online so nodes can quickly authenticate and negotiate a shared symmetric key.

2.3.4 Antigone

The security architecture setup by McDaniel [McDaniel] gives a full set of security mechanisms that can be chosen and combined to provide the policies desired for the group. There are mechanisms for joins, leaves, failure detection, authentication, application messaging, and group rekey/membership. The most interesting of these is the

rekey/membership mechanism. The mechanism is a joint message that has the new key attached with the membership. This is shown in (a) of Figure 5. The joint message is encrypted for each receiver separately, concatenated, and sent out as one large rekey message to the entire group. Figure 5 shows a full rekey message for Antigone in (b).

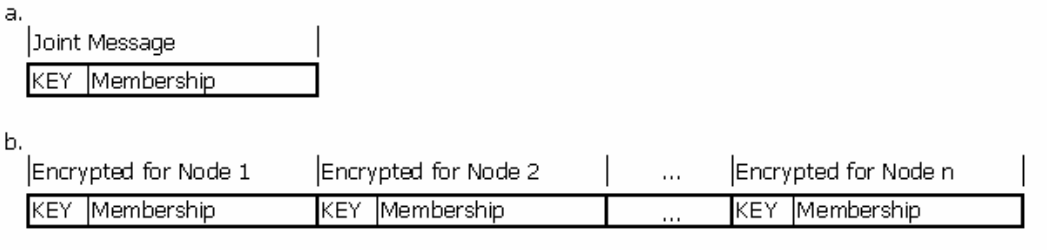


Figure 5. a. Antigone Joint Message b. Full Antigone Rekey Message

Rekeying the group using this concatenated joint message does not effectively scale to large groups. A joint message grows linearly with the size of the membership. In addition, the number of joint messages that are concatenated grows linearly with the size of the membership because a joint message is encrypted and concatenated for every member. Thus, the size of the rekey message grows exponentially.

2.3.5 Secure Channel Caching

The SecChan layer [Rodeh] provides secure unicast channels. The layer keeps negotiated symmetric keys associated with current group members cached to avoid the expense of reestablishing the secure channel. Channels are dropped from the cache if they are no longer associated with a current group member or if they are still active after a set expiration period to prevent cryptanalysis. This process of caching symmetric secure channels avoids the cost of establishing new connections at the expense of storage.

2.3.6 Gossip using Catalog Messages

A gossip-based network, like the anti-entropy phase of bimodal multicast [Birman], is designed for intermittent links. Node A selects another node at random, Node B, and sends a catalog of recently received messages to the node, as in Figure 6. Node B checks to ensure it has received all messages in the catalog, and requests any messages in the gossiped catalog that it has not already received. Node A then sends the requested messages.

Gossip relies on the random spreading of messages in lieu of a brute-force method of total delivery. Thus, a gossip network adapts to intermittent links keeping live nodes updated and attempting to keep ill nodes updated as much as possible.

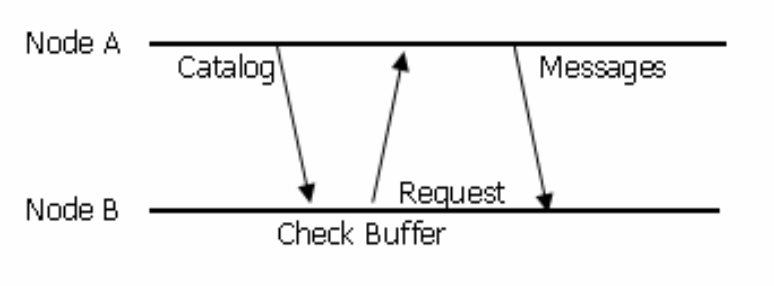


Figure 6. A single gossip

Gossip also avoids logical single points of failure. The random selection of nodes does not limit logical groups to be singly connected at any point although it is possible that single points of failure remain, at the physical level.

2.3.7 Gossip as a Self-Sufficient Protocol

Gossip [Birman] is a part of a larger protocol. Gossip ensures that all of the nodes in a group have the same messages. Gossip has since been modified to be the entire message distribution scheme. As the sole message distribution method, the likelihood of

a node not having all messages increases. This coupled with the likelyhood of small messages motivated changing gossip to send the actual messages in place of the catalog. This eliminates a receiving node sending a request for messages as well as the sending node sending a response which cuts the load of new messages from three transmissions to one.

2.4 Summary

This chapter discusses current research in group communication and security protocols. Message keys, secure channel caching, joint key/membership messages, and gossip networks from this chapter are used in the following chapters to introduce a novel approach to key distribution in mobile ad-hoc networks.

III. Methodology

3.1 Problem Definition

MANETs are inherently less reliable than wired networks. The ability of nodes to achieve the connectivity and bandwidth that would be possible if no link failure or interference existed, can vary greatly in a group. Mobility alone causes significant connectivity problems. A node may be experiencing interference due to neighboring transmitters. Moving behind an obstruction such as a mountain or building, could cause a drop in signal.

Reliable group communication protocols often expend network resources to improve reliability. Probabilistic multicast protocols, like gossip, expend resources to keep healthy nodes updated. Unhealthy nodes are updated when possible, but these nodes may not have the same level of consistency that they would have with a reliable multicast protocol. The goal of a gossip protocol is to increase availability by ensuring that "healthy" nodes receive updates with high probability. "Unhealthy" nodes communicate as well as they are able and do not cause the failure of "healthy" nodes, in probabilistic multicast systems like gossip.

A security protocol for mobile wireless communication can be built on similar principles as the gossip communication protocol to increase system availability at the cost of inconsistency in unreliable nodes as well as match the expected losses, faults, and join/leave events in wireless environments. With gossip, that means that the security protocol must expect a highly dynamic environment and should not be greatly affected by

unhealthy nodes. The offered load of the security protocol should be distributed across the network to increase system scalability.

3.1.1 Goals and Hypothesis

This research improves the scalability of a secure mobile ad-hoc network. By using concepts from catalog-based gossip networks [Birman], the load of key and message dissemination is dispersed among the members of the group thereby reducing the impact of intermittent node failure on the scalability of the network.

3.2 Approach

By combining established ideas, this section proposes a novel key distribution protocol for mobile ad-hoc networks. The general idea is similar to the catalogs used by initial versions of gossip with the actual messages acting as catalogs of the required keys. The cached unicast security channels of SecChan [Rodeh] reduce computational delays and key negotiation delays between nodes. The combination forms a powerful protocol, which maintains network security with relatively low overhead and has high resilience to environments prone to churn like mobile networks.

3.2.1 Secure Unicast Links

It would be possible to avoid key distribution in gossip by using unicast encryption schemes for each gossip message. In the unicast encryption scheme, a sending node encrypts a gossip message before sending the message to the receiver. The receiver then decrypts the message before it can be delivered or gossiped. Establishment of the unicast channel must also be included in this process. In addition to the inherent

delay in this process, there is no way to account for the membership of the group which must be attached to every message inside of a gossip message. Consider the situation in Figure 7.

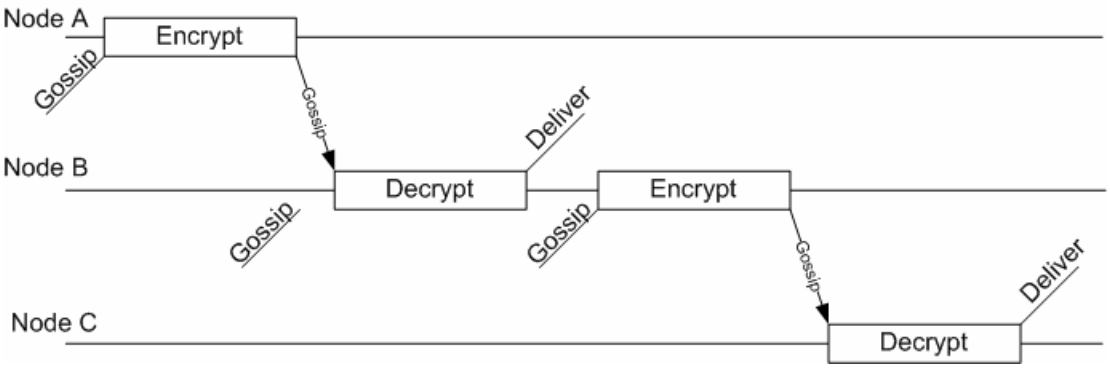


Figure 7. A single message being gossiped without message keys

Node A is sending a gossip message to node B. Node A must first encrypt the message with the key for node B. While node A is encrypting the message, node B decides to gossip to node C. The gossip to node C does not contain the message from node A because it has not been sent yet. In fact, the message will not be included in gossip messages from node B until after the message has been delivered to node B. At some point after the message has been delivered to node B, node B again decides to gossip to node C. The message is now included in the gossip message. As before, node B must encrypt the message with the key for node C before the message can be sent and must be decrypted before the message can be delivered or gossiped.

3.2.2 Minimize Encryption

Similar to Iolus, the originator of a message is like the group security controller (GSC) for the group and other nodes in the group are like group security intermediaries

```
1  Send()
2    Generate message key
3    Encrypt message with key
4    Attach message key ID to encrypted message
5    Add message to buffer
5  End
```

Figure 8. Send procedure

```
1  Receive()
2    Receive encrypted message buffer
3    For (messages in new buffer)
4       If (message not in current buffer)
5          Add encrypted message to current buffer
6       End-if
7       If (message key not in key cache)
8          Send message key request
9       End-if
10   End-for
11 End
```

Figure 9. Receive procedure

(GSI). The send procedure using pseudo-code is shown in Figure 8. The originator creates a symmetric key to be used as the message key, K_m. The originator encrypts the message (M) using the K_m giving $e(M,K_m)$. A key identifier is also added to the encrypted message which consists of the address of the originator and a unique identifier assigned by the originator. There are no duplicate key identifiers are due to the inclusion of the originator's address. Now $e(M,K_m)$ + KeyID can be included in gossip messages

```
1  Deliver()
2    Decrypt Message with Message Key
3    Deliver Message
4  End
```

Figure 10. Message Delivery

without needing further encryption. The receive procedure in Figure 9 does not contain message decryption as the decryption task is delayed until message delivery as shown in

Figure 10. Encrypting the message with a message key prevents the message from being reencrypted by nodes in the group, reducing the amount of data that must be encrypted and decrypted to only the message key.

```
1  Key Received()
2     Decrypt Key
3     If (Key not in key cache)
4       Add Key to message key cache
5     End-if
6  End
```

Figure 11. Receiving a message key

A node receiving a gossip message must determine by examining the messages contained in the gossip which keys it needs. The node then sends a request for the needed keys to the originator of the gossip message. The sender replies with the requested keys over a secure channel as shown in Figure 12 to be processed by the requester as in Figure 11.

```
1  Key Request Received()
2     If (message key not in key buffer)
3        Ignore key request
4     Else
5        If (request node is not in key ACL)
6           Send key request failure
7        Else
8           If (secure channel is not in cache)
9              Establish secure channel
10          End-if
11          Send key via secure channel
12       End-if
13    End-if
14 End
```

Figure 12. Receiving a key request

In Figure 13, a gossip message sent immediately following node A gossiping to node B. Node B receives the message and adds the message to the buffer of messages to

be gossiped. This is possible because the message is still encrypted with the message key and must be encrypted such to be gossiped. Node B must then request the message key from node A. Differing from the previous example, when node B decides to gossip to node C, the message is included in the gossip even though node B has not yet received the message key or delivered the message. Thus, node C receives the message much quicker. Node C then requests the message key from node B.

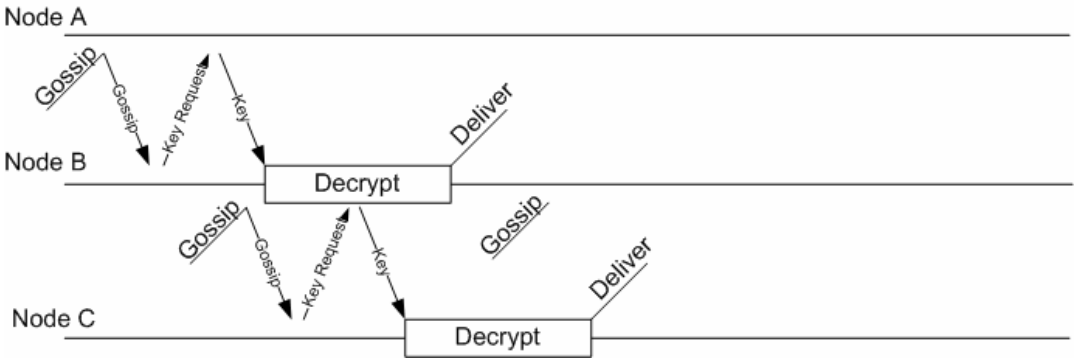


Figure 13. A single message being gossiped with message keys

3.2.3 Membership

As seen in Figure 12, a node that forwards a message key must know who is allowed to receive the key. Therefore, to establish the membership of the group to which the message is sent, the originator attaches the access control list (ACL) to the message key, similar to Antigone's joint key [McDaniel]. Thus, each key has the group membership attached. This has increases the amount of data that must be reencrypted at each node to the size of K_m and that of the ACL.

A node might not have received a key that has been requested by another node. Consider a node, N_B, that receives a gossip message and send a request for a message key

that it does not have to N_A. N_B may gossip the same message to another node, N_C, which also does not have the required message key. N_C could send a key request to N_B before N_B receives the message key it requested before. In this case, N_B drops the key request and does not respond to it. N_C interprets this as a lost message and will try the request again. Recall that to overcome broken links, N_C will probabilistically choose to request the message key from either N_B or the originator of the message key.

3.2.4 Caching Secure Unicast Channels

The gossip protocol relies heavily on secure unicast channels between nodes. Uunicast channels have the benefit of being usable across overlapping groups so two nodes belonging to the same groups can use the established secure unicast channel for communication requiring such a channel in all groups they share membership. For example, nodes N_A and N_B are members of both group G_1 and group G_2. They only need to establish a channel between them one time to allow sending of secure data for both groups G_1 and G_2.

To further optimize the secure unicast channels, public key encryption can be avoided by using a method similar to Rodeh's SecChan layer [Rodeh]. A node pair negotiates a symmetric link key (K_L) for sending secure data. Each node keeps a cache of negotiated symmetric keys so the pair does not need to renegotiate a new symmetric key for each use of a secure channel. This reduces the processing involved in reencrypting message keys because the decryption and encryption both are done using less expensive symmetric processes.

The separate link encryption by each node pair takes advantage of the physical attributes of different links. A node might handle the unicast link encryption differently over laser link than it would over an RF link. Since a sender transmits the message key over a laser link, encryption may be deemed unnecessary as the probability of interception is extremely low. This eliminates the processing expense of encrypting the message key on the sender and decrypting the message key on the receiver.

3.3 System Boundaries

The system under test consists of the simulated environment in which the scenarios are run. Included are the security and transmission protocols as well as simulated mechanisms to send, transport and receive messages. The distribution schemes, Iolus and gossip with on-demand key distribution, make up the component under test.

3.4 System Services

The system provides secure delivery of data to a group of nodes. The system provides key distribution to accomplish this end. The system is fully successful when a message is delivered to all members of a group. For MANETs, this may be more appropriately restricted to delivery to all the healthy nodes of a network. The system fails when nodes cannot deliver a message. This can be due to either not receiving a message intended for the node or not receiving the key needed to decrypt the message so that it can be delivered. The system also fails when a message is delivered by a node for which

the message was not intended. This requires that nodes receive both the encrypted message and the key or that a node sends a plaintext message.

3.5 Workload

The offered load is the number of sent messages per second, 20 messages per second. Messages to be delivered to all nodes in the group must be sent securely

3.6 Performance Metrics

- Percentage of Messages Delivered – This is the percentage of group members that received a particular message. When comparing a probabilistic multicast protocol to a reliable protocol, the impact on message delivery is critical.
- End-to-end delay – This is the time from when the message is submitted to the system to the time that a message and its key is received at a node. This is the amount of time that a message is in the system.
- Load – This is a measure of the load on nodes in the network in terms of the number of messages (or bytes) per unit of time. Of particular interest is the maximum load and load standard deviation which indicates how well the load of the network is distributed to the group.

3.7 System Parameters

- Number of groups/senders – The number of groups affects how nodes interact with each other and the amount of data entering the network.
- Topology – The network topology determines the amount of work required to transmit data. The topology for all scenarios is set up as shown, or is similar to

Figure 14. For 101 node scenarios, there are 20 nodes at each level. For 2 group scenarios, the topology still has 20 nodes at each level, but also has an additional sender. Each group contains 12 nodes from each level with the middle 4 nodes at each level in both groups.

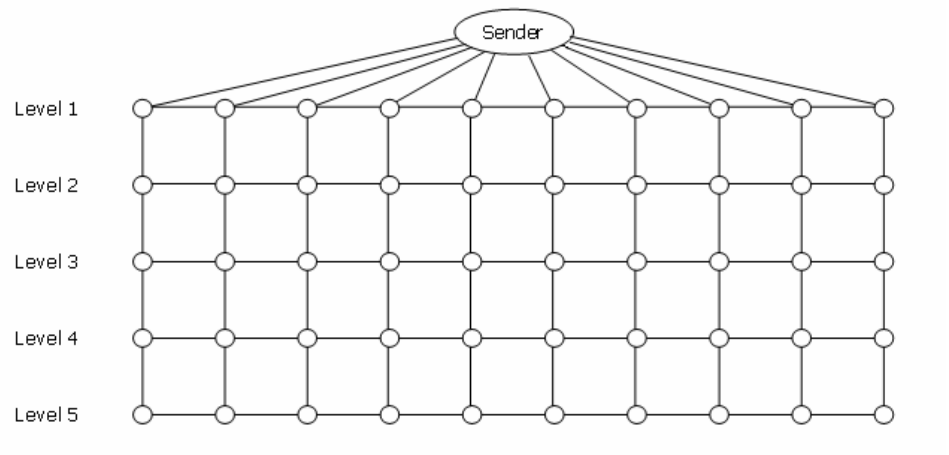


Figure 14. 51 node mesh topology

3.8 Workload Parameters

- Node Failure Rate – The number of failures per unit of time determines how much work the network must do to compensate
- Group Size – The group size affects how widely a message must be distributed.
- Message Rate – The rate that messages enter the system directly affects the load placed on the system.

3.9 Factors

Each of the scenarios in Table 1 is run as a simulated Iolus distribution system and then again with a gossip message distribution system with an on-demand key distribution. The key distribution scheme replicates a workload as it might be

experienced in a mobile ad-hoc network, and the factors were chosen in consideration of such an environment. The node failure rate is likely the most important factor due to its prevalence in MANETs. Other important factors include group size and the number of groups. Table 1 lists the settings for each of the factors.

Table 1. Factors Settings

Scenario	Messages per second	Nodes	Groups	Group Size	Node Failure Rate
1	20	51	1	51	0%
2	20	51	1	51	3%
3	20	51	1	51	10%
4	20	101	1	101	0%
5	20	101	1	101	3%
6	20	101	1	101	10%
7	20	102	2	61	0%
8	20	102	2	61	3%
9	20	102	2	61	10%

3.10 Evaluation Technique

The key distribution protocol described for gossip and Iolus use network simulator 2 (NS2). NS2 is a network simulation program written in C++ and TCL [Breslau00]. NS2 simulates communication between nodes for a user-defined topology which are set up using TCL scripts.

3.11 Validation

There currently are no published results for the Iolus protocol to validate against. However, distinct load characteristics are anticipated for Iolus results. The load is expected to focus on a small set of nodes, the GSAs. This will give high maximum load values relative to the mean load per node.

3.12 Experimental Design

The experiments include the 2 message distribution schemes (Iolus and gossip with on-demand key distribution), 3 node fault rates, 3 group configurations, and 10 trials. This provides 18 experiments of 10 trials each giving a total of 180 experiments.

3.13 Summary

This chapter describes an experiment to analyze the effectiveness and scalability of message distribution schemes for mobile ad-hoc networks. Measuring the impact on end-to-end delay and the load of varying node failure rates allows an assessment of the scalability of the distribution scheme. Measuring the percent of nodes receiving the messages determines at what cost that scalability comes.

IV. Results and Analysis

4.1 Chapter Overview

This chapter compares an Iolus-based distribution to a gossip-based distribution with on-demand key distribution using the scheme from Chapter 3. The end-to-end delay and load on the network are used to analyze the scalability of each system and the percentage of messages received is used to determine the cost of using an unreliable protocol. All scenarios were run for 300 seconds with a 120 second fault period starting at 123.5 seconds. All charts truncate the first 100 seconds and last 40 seconds. Additional charts not mentioned in Chapter 4 can be found in Appendix A.

4.2 End-to-end Delay

The first metric considered is the end-to-end (ETE) delay of each system. Each message was timed from entering the system, prior to any encryption, to being delivered at each node which means the node must have both the encrypted message and the message key.

Figure 15 shows the ETE delay for the 51 node gossip scenarios. Each of the three scenarios has a different node failure rate. From the chart, average ETE delay increases as the node failure rate increases. With a node failure rate of 3%, the average ETE delay increases about 5% on average. Increasing the node failure rate to 10% results in an increase of about 15% in the average ETE delay.

The Iolus baseline (0% node fault rate) scenario has a much lower ETE delay than the gossip baseline. However, Iolus is more sensitive to faults. This is also true for the

101 node scenarios shown in Figure16. The increased workload due to the number of nodes causes the average ETE delay for Iolus with the 10% node fault rate to exceed that of the gossip equivalent. The 2 group scenarios, in Figure 17, see a better ETE delay for Iolus likely due to the increased number of senders. However, the delay has a similar sensitivity to the node fault rate.

Table 2 gives a summary of the percent increase in ETE delay for each of the runs. This is the percent increase in the average delay over the 120 second node failure time compared to the 0% node failure rate scenario. The 3% node failure rate gives between a 59% to 83% increase in Iolus ETE delay, and the 10% node failure rate causes a nearly 400% increase in the 101 node scenario. The gossip scenarios see a relatively constant 5% increase for a 3% node fault rate and about 15% increase in ETE delay for the 10% node fault rate scenarios.

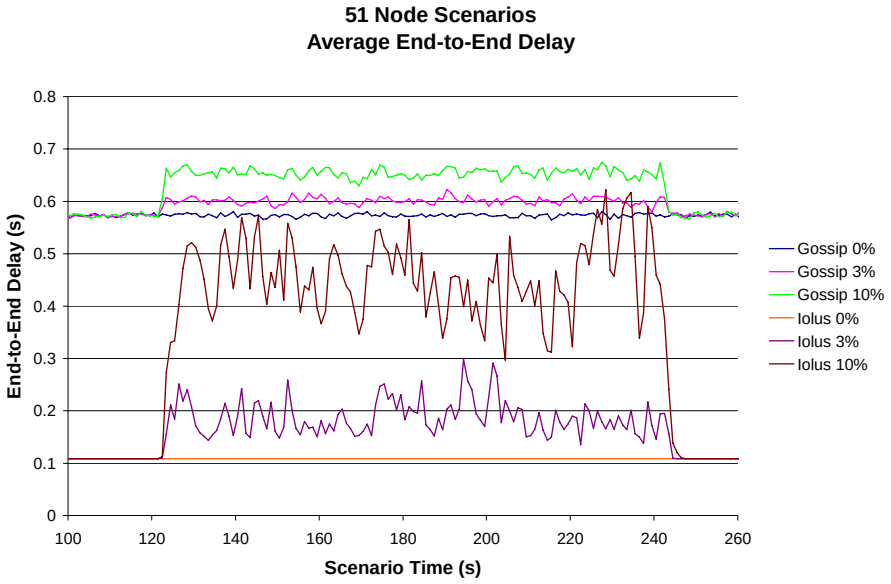


Figure 15. Average End-to-End Delay for 51 node scenarios

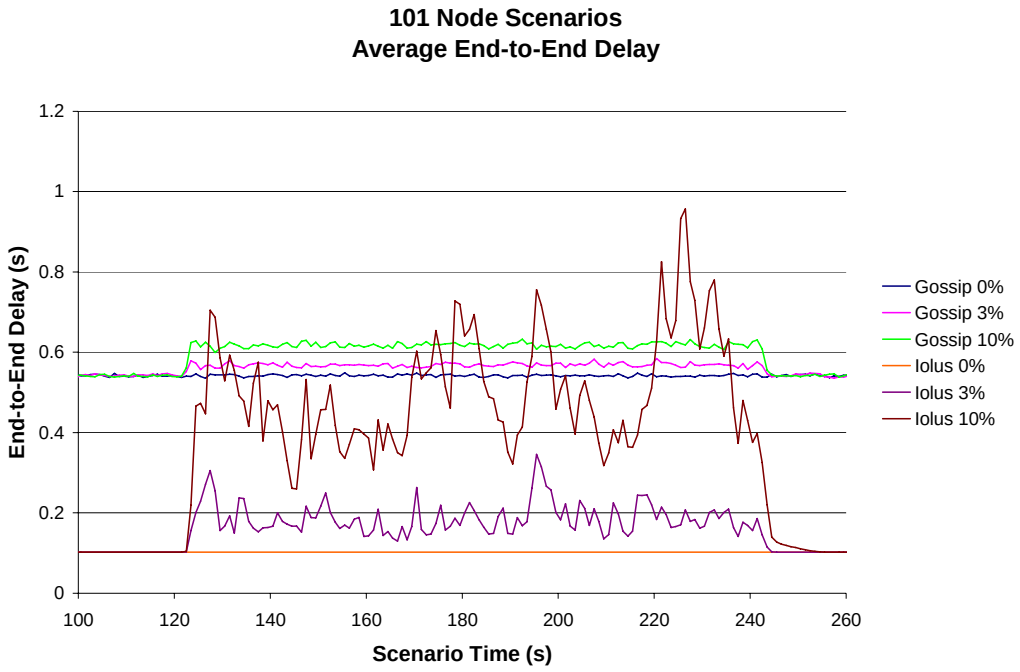


Figure 16. Average End-to-End Delay for 101 node scenarios

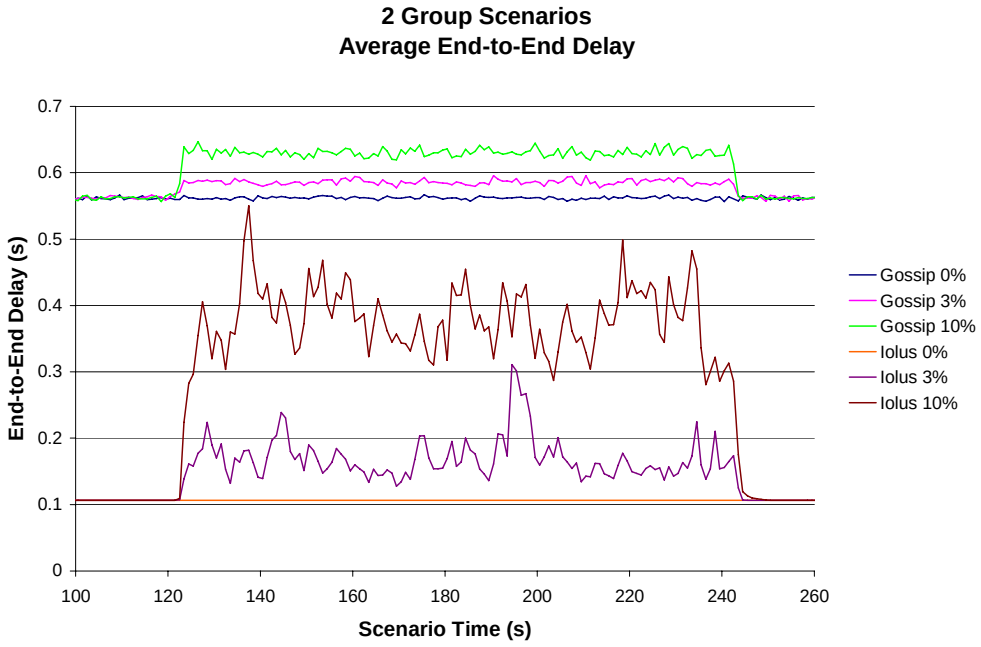


Figure 17. Average End-to-End Delay for 2 group scenarios

Table 2. Node Failure Rate and End-to-End percent increase for all scenarios

Group	Node Fault Rate	Iolus	Gossip
51 Nodes	3%	73%	5%
	10%	318%	14%
101 Nodes	3%	83%	5%
	10%	390%	14%
2 Group	3%	59%	4%
	10%	254%	12%

A similar trend appears in the standard deviation for the end-to-end delay in the 51 node scenarios as shown in Figure 18. The standard deviation slightly decreases for gossip likely due to the elimination of outliers by message expiration. The standard deviation for Iolus continues to increase as the node failure rate increases. Charts for the ETE delay standard deviation for 101 node scenarios and 2 group scenarios display similar characteristics and can be found in Appendix A.

Figure 19 shows the maximum ETE delay for the 51 node scenarios. For the 10% node failure rate, the Iolus maximum ETE distribution is as high as 38 seconds. The maximum ETE delay for gossip stays at about 2 seconds. 101 node scenarios and 2 group scenarios also exhibit this high maximum ETE delay shown in Figures 20 and 21, respectively. The maximum delay for Iolus peaks as high as 30 seconds in the 101 node scenarios and 25 seconds for the 2 group scenarios. Gossip maximum end-to-end delay remains at 2 seconds for both sets of scenarios

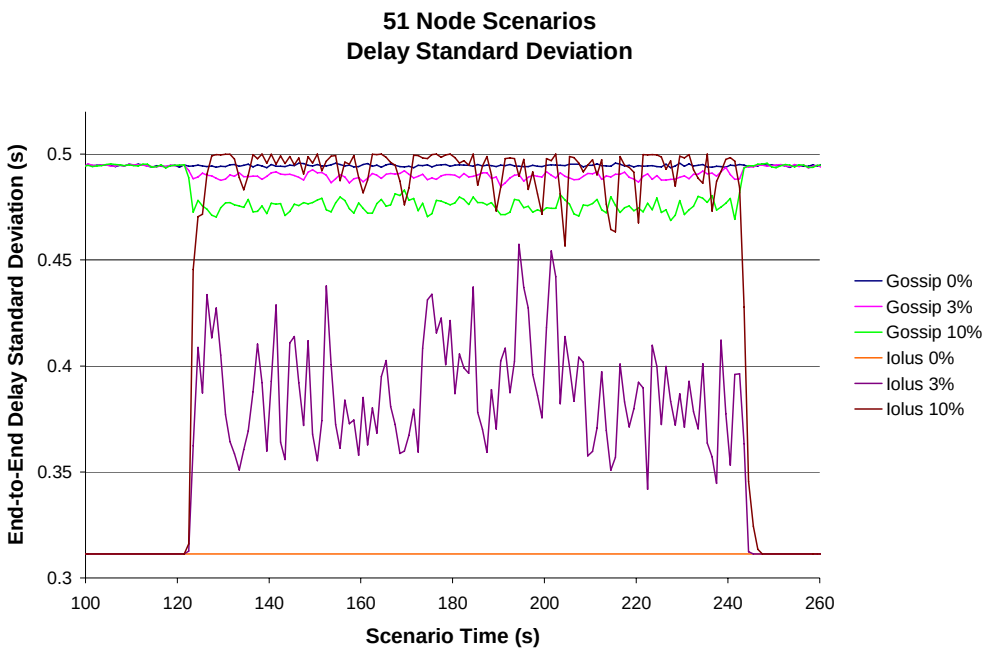


Figure 18. End-to-End delay standard deviation

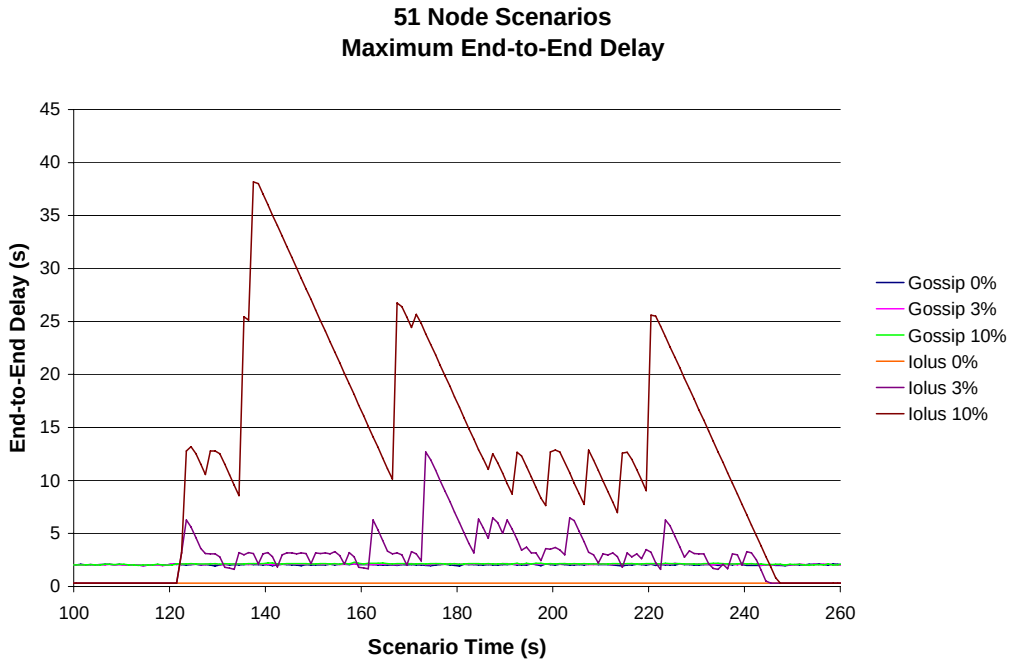


Figure 19. 51 Node Scenarios Maximum End-to-End delay

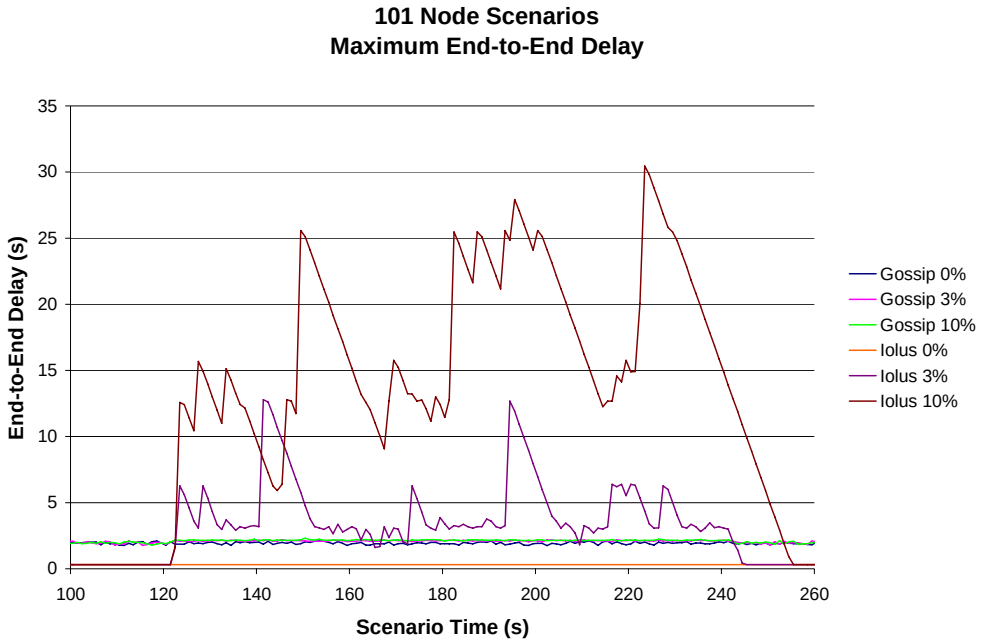


Figure 20. 101 Node Scenarios Maximum End-to-End delay

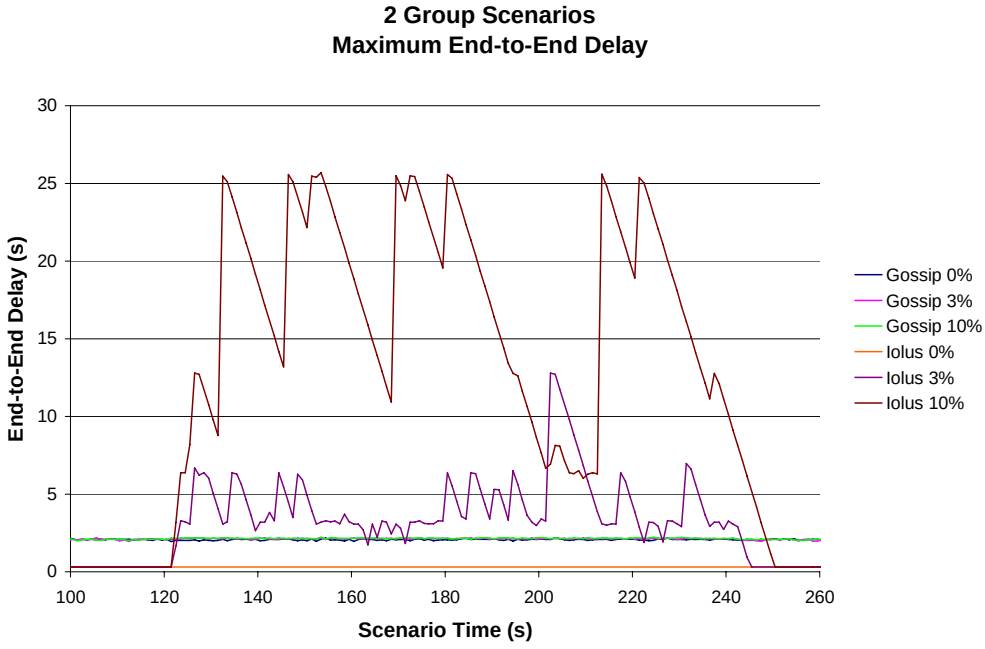


Figure 21. 2 Group Scenarios Maximum End-to-End delay

Figure 22 compares the average time to distribute the message key for each of the 51 node scenarios. Gossip has very minimal increase in distribution times. The key distribution time increases from about 150ms for 0% node fault rate to 200ms for 10% node fault rate. Iolus has a lower key distribution time averaging 100ms for the 0% node fault rate case. The Iolus key distribution delay is on par with gossip at the 3% rate. However, Iolus key distribution is greatly affected by the 10% node fault rate reaching as high as 700ms. Charts for the remaining scenarios can be found in Appendix A. Figures 23 and 24 summarize the group types giving the average and maximum key distribution delays for the 10% node fault rate scenario for each group type. The gossip key distribution scheme consistently achieves lower key distribution delays than the Iolus-based scheme under higher node fault rates.

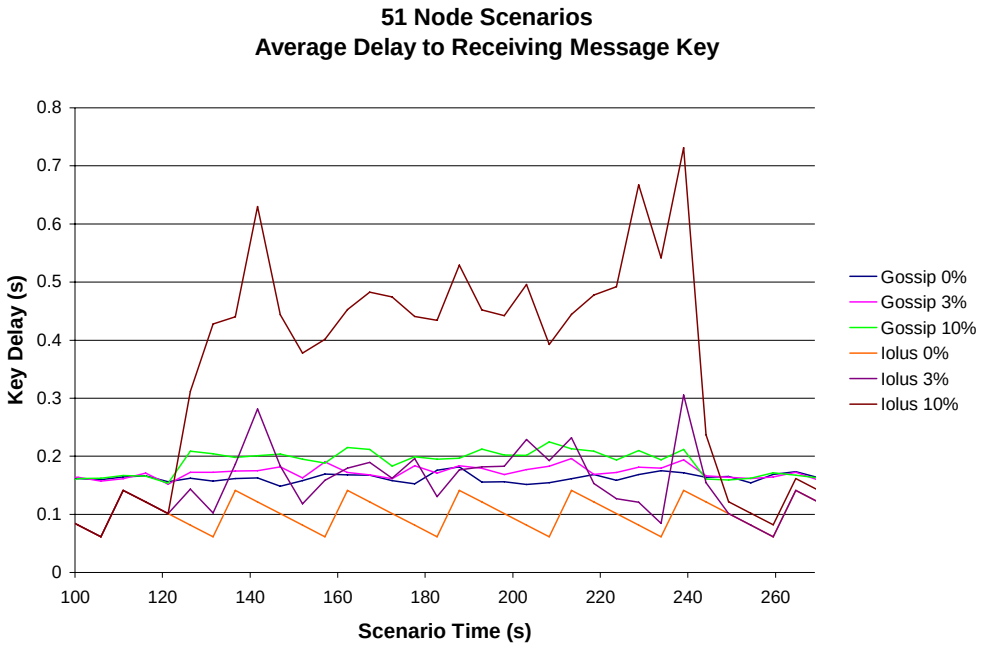


Figure 22. Average key distribution End-to-End delay

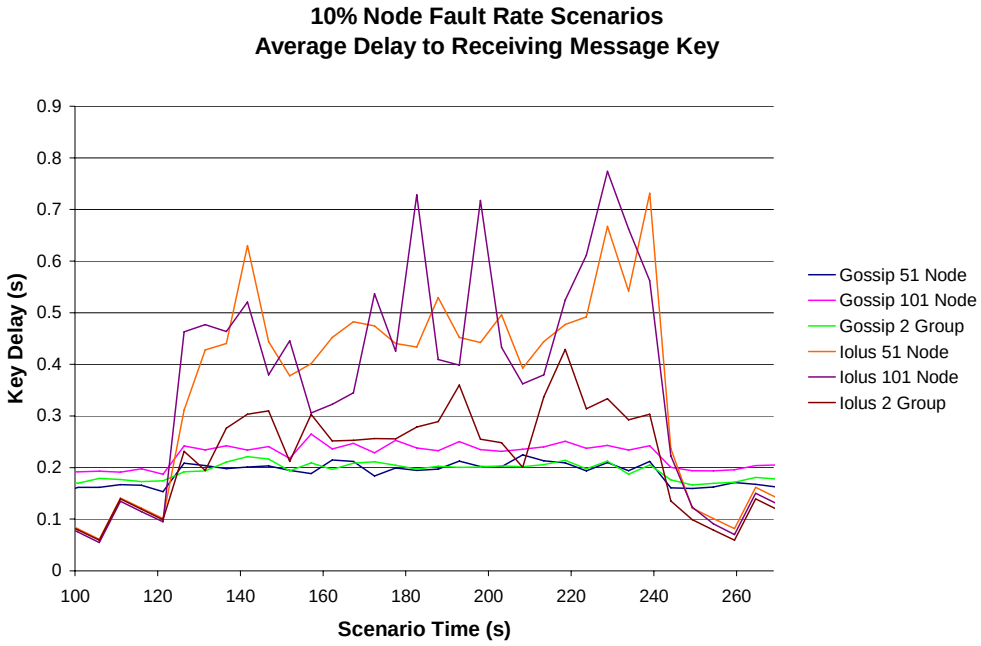


Figure 23. Average message key distribution End-to-End delay for 10% node fault

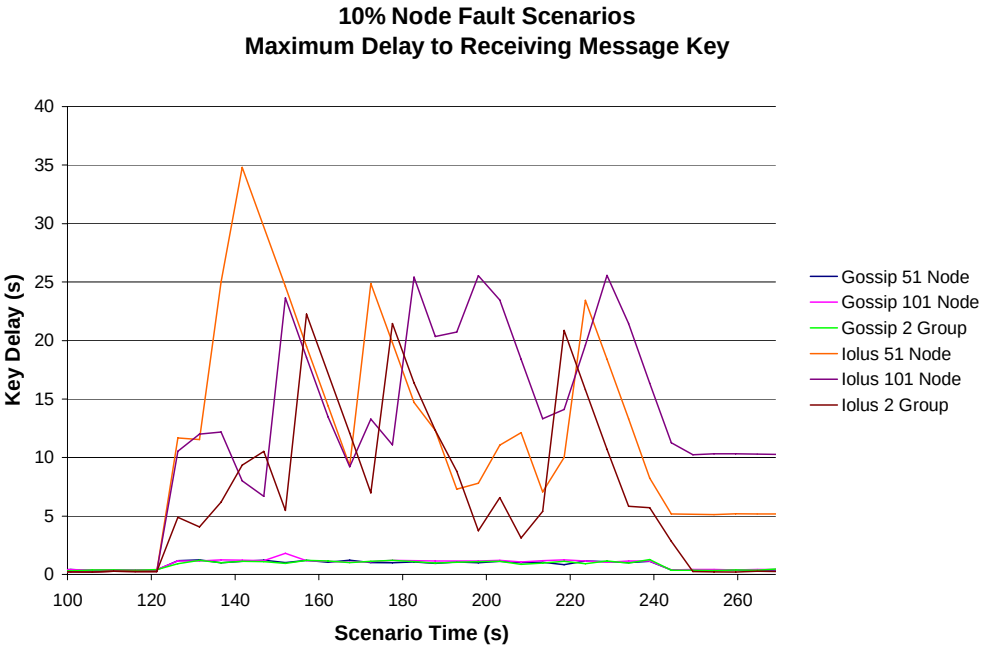


Figure 24. Maximum message key End-to-End delays for 10% node fault

4.3 Network Load

The network load shows how the work is distributed to all of the nodes on the network. Figures 25 and 26 present the average and maximum message send rate for each of the 51 node scenarios. The average for each of the gossip was a constant six messages due to a constant, non-adaptive gossip rate. In Figure 26, we see that the message send rate extends as high as 500 messages per second and even the base is greater than the constant gossip rate.

Figures 27 and 28 show the average and maximum send rates in bytes sent per node. The average send rate in bytes in the gossip scenarios range between 90,000 and 100,000 bytes/s. This is much higher than the average Iolus send rate. However, as with the maximum message send rate, the maximum send rate in bytes for Iolus varies greatly and is frequently higher than the send rates of gossip. At one point, the maximum sent bytes of Iolus is about double the maximum sent bytes of gossip.

While the average case for Iolus appears to give a lesser load on the network, there are a handful of nodes that are doing all of the work. This can be seen in the comparison of the average send rate in Figure 25 to the maximum send rate in Figure 28. Gossip generally distributes more data than Iolus, but this load is spread amongst all of the nodes in the network. In addition, gossip uses a smaller amount of larger messages to distribute data due to the use of message buffers.

Also of interest in Figures 24-30, the gossip scenarios exhibit little to no change in send rate when the node fault rate is increased. The increased node failure does not increase overhead load on the network.

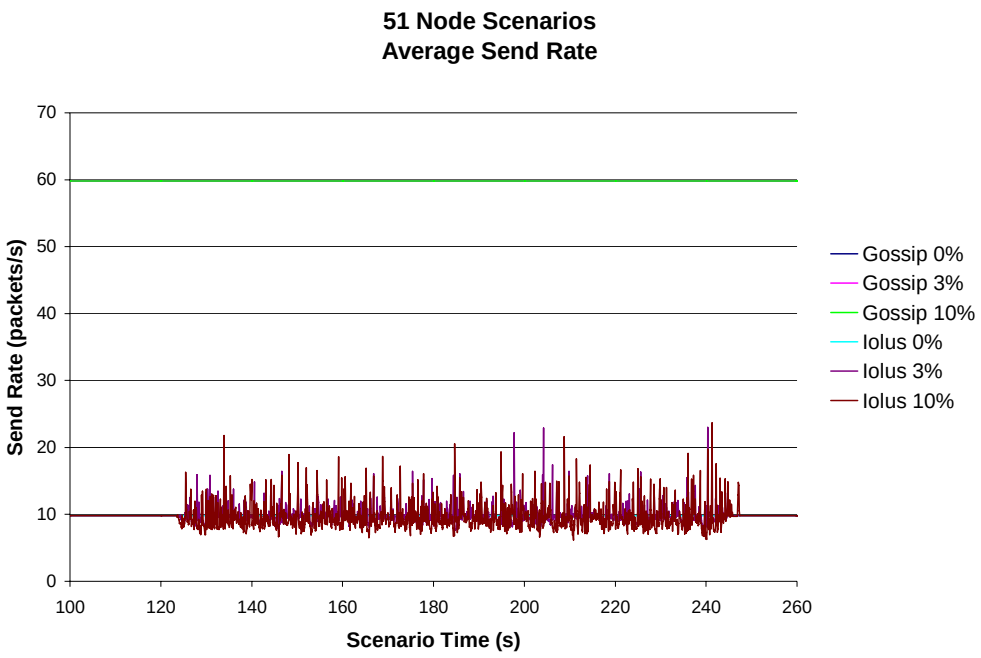


Figure 25. Average message send rate

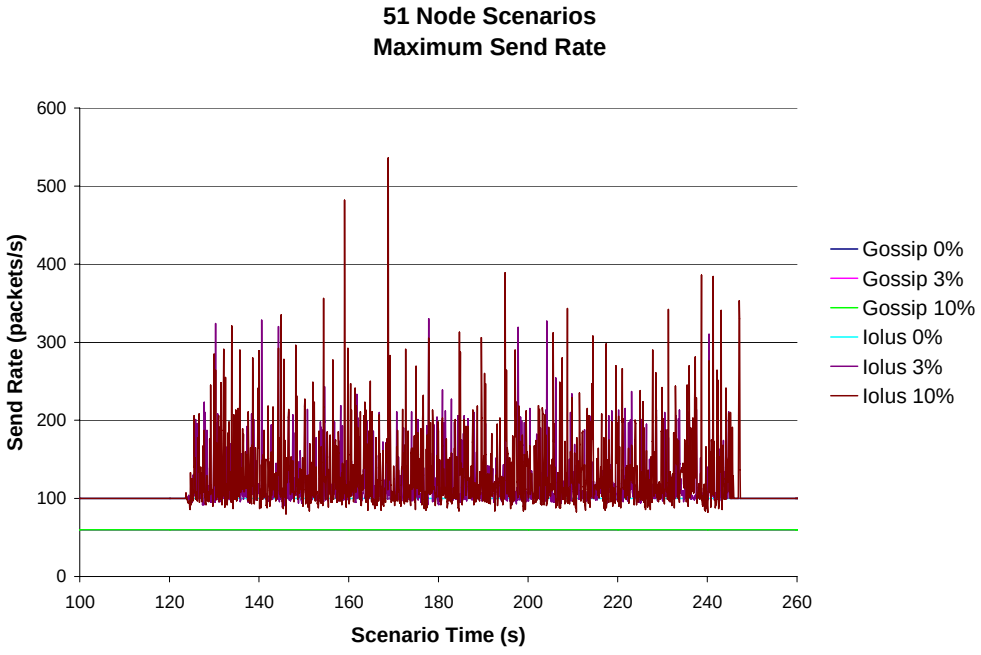


Figure 26. Maximum message send rate

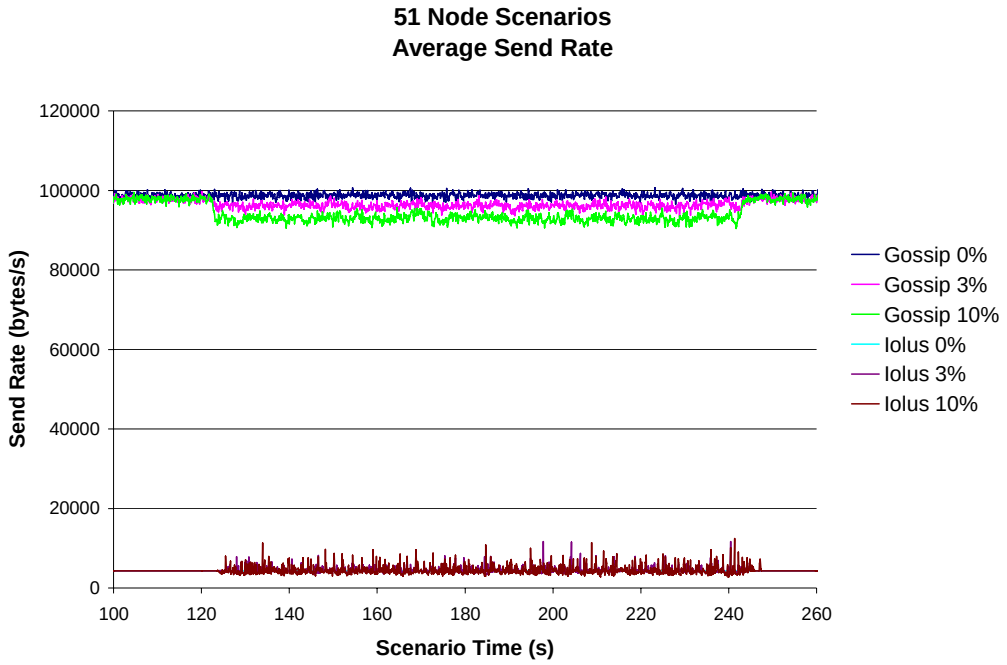


Figure 27. Average send rate per node

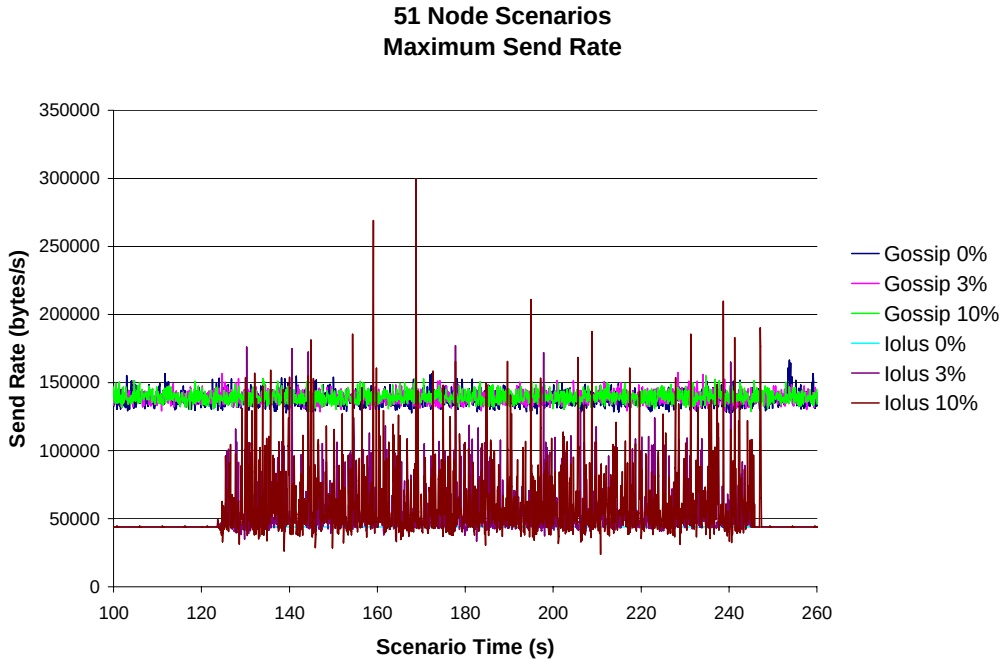


Figure 28. Maximum send rate per node in 51 node scenarios

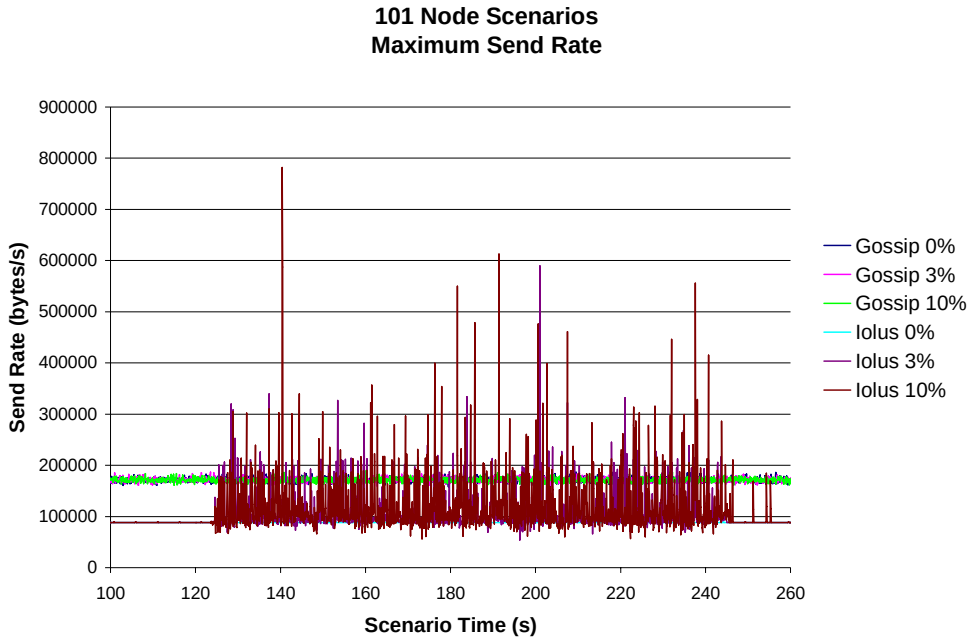


Figure 29. Maximum send rate per node in 101 node scenarios

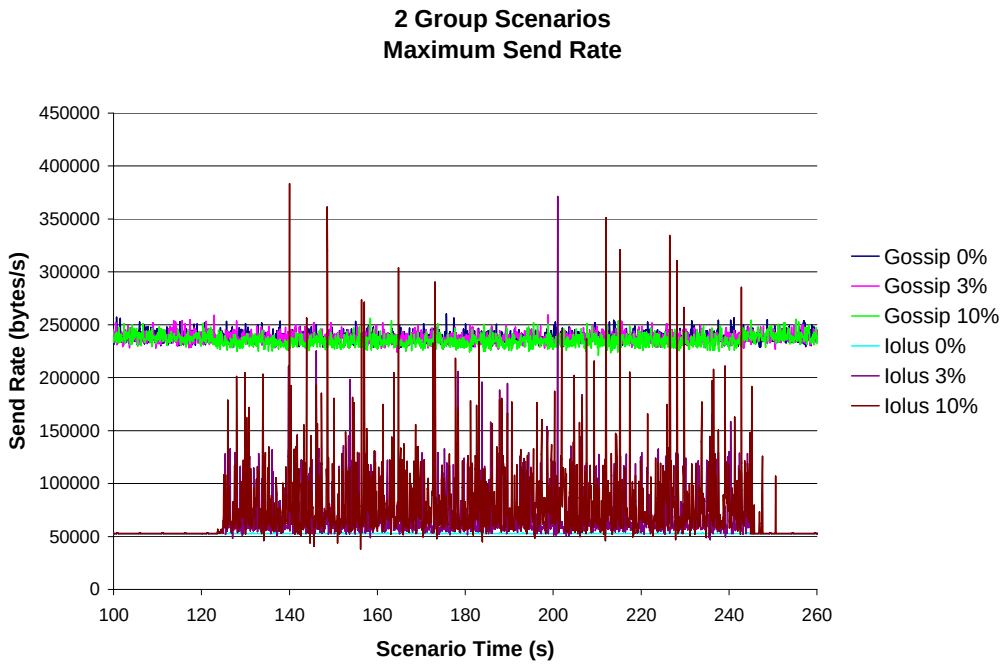


Figure 30. Maximum send rate per node in 2 group scenarios

4.4 Percent of Messages Delivered

The cost of unreliable communication is analyzed by looking at the percent of messages that were not delivered. Figures 31, 32, and 33 show this for each of the gossip scenarios. This is not given for Iolus as all communication is reliable (i.e., re-sent until acknowledged). The 0% node fault rate for gossip gives average reliability above 99% for all group types. A node fault rate of 3% lowers the reliability to 98-97%. The 10% node fault rate case goes as low as 91%. This rate is influenced by the message expiration time in gossip. If desired, higher reliability could be achieved by lengthening this expiration time.

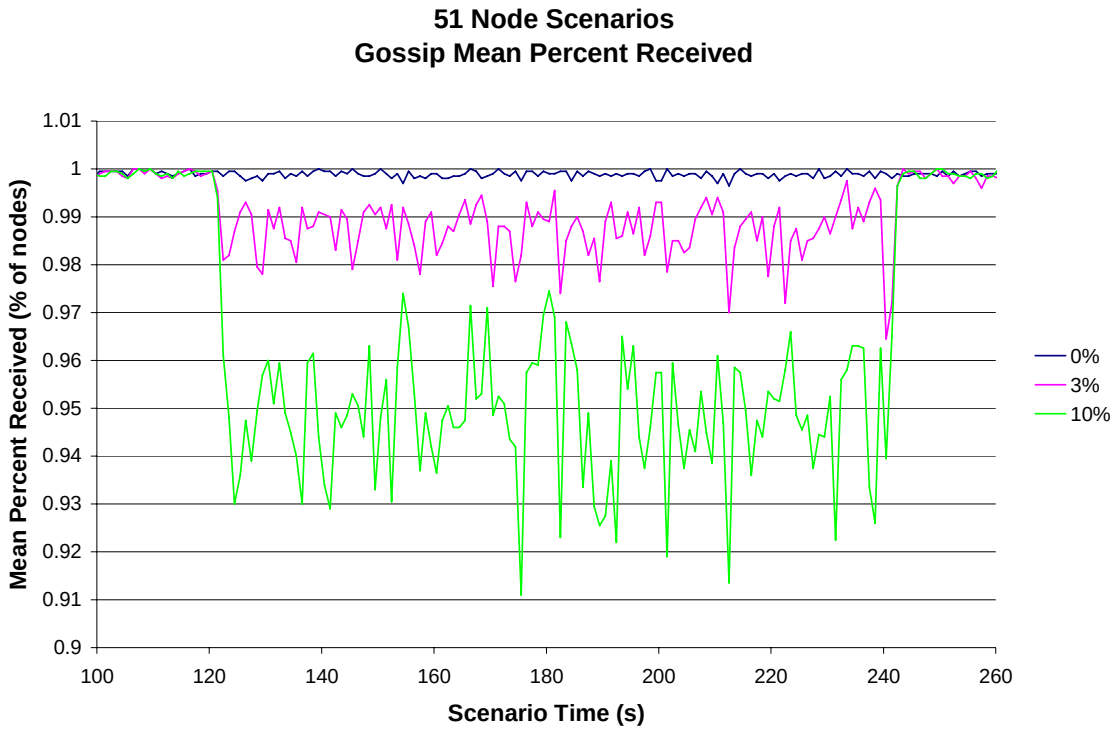


Figure 31. Percentage of gossip messages received in 51 node scenarios

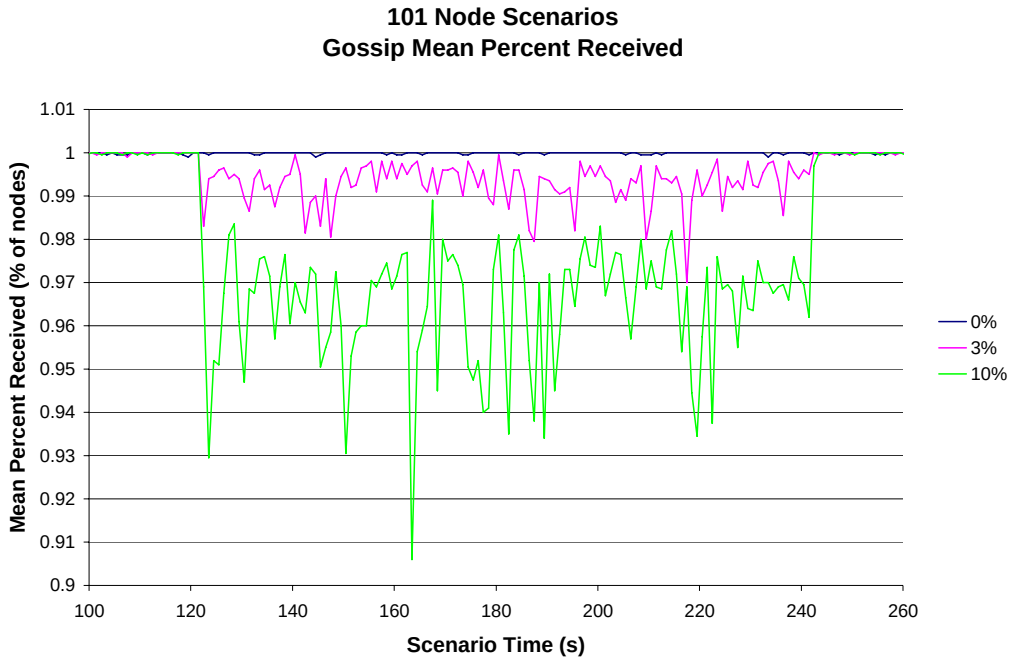


Figure 32. Percentage of gossip messages received in 101 node scenarios

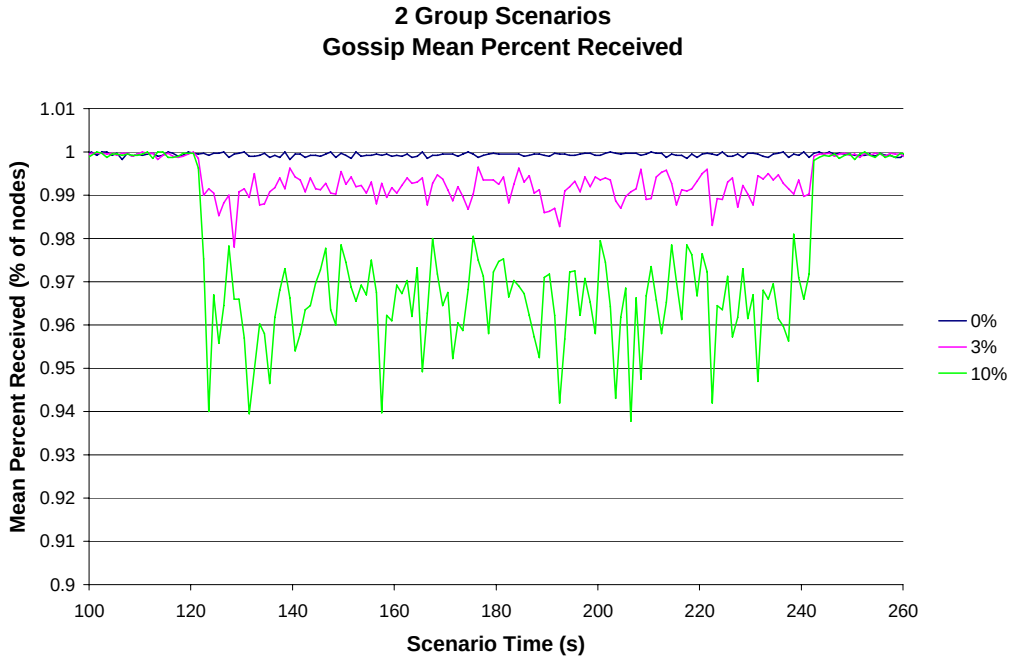


Figure 33. Percentage of gossip messages received in 2 group scenarios

4.5 Summary

This chapter analyzes the results from the experiment setup in Chapter 3. The gossip system has a reliability of over 90% for all scenarios. This compares to the Iolus system that was designed to give 100% reliability.

While Iolus performs better in the stable scenarios, gossip outperforms Iolus in for increasing node failure rates. Iolus experienced ETE delays over 35 seconds for a 10% node failure rate for both key and message distribution. Gossip handled node failure rate consistently across group types. The 10% node failure rate gave only a 15% increase in average ETE delays. Under the same failure rate, Iolus displayed a 250%-400% increase in ETE delay.

The Iolus system has a lower average network load, but appeared to have hotspots where the load was drastically higher than the rest of the group. Gossip had a higher average load, but had a much lower difference between average and maximum load per node.

V. Conclusion

5.1 Chapter Introduction

The typical working environment of MANETs often consists of frequent node outages, unreliable links, and open communication. Transmitting secure data in this environment calls for a scalable, low-overhead key distribution scheme that delivers messages quickly to healthy nodes while coping with unhealthy nodes. Gossip provides a high-probability of delivery basis to base such a scheme upon. This research proposes and analyzes on-demand key distribution based on gossip.

5.2 Conclusion

Gossip with on-demand key distribution appears to scale well in unreliable networks. Gossip, under a 10% node failure rate, gave only a 15% percent increase in average ETE delay compared to Iolus resulting in 250%-400% percent increase in ETE delay. Message keys, secure channel caching, and joint keying distribute the key and message distribution workload while incurring little additional overhead. Gossip with on-demand key distribution distributes messages effectively in accord with the node's ability to receive.

The on-demand key distribution scheme and gossip tolerates an increasing node fault rate, a characteristic typical to MANETs, while distributing the load of message and key distribution to all of the nodes in the group. The maximum send rate for gossip stayed at 140 kB/s, while Iolus peaked at 300 kB/s.

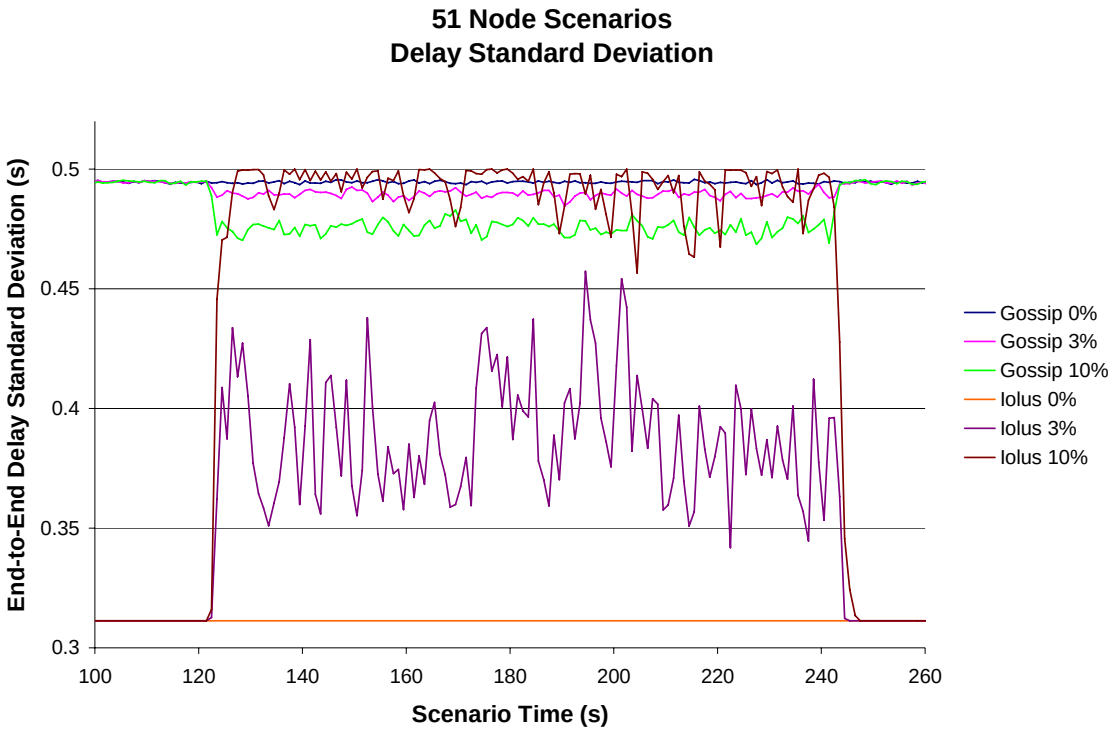
5.3 Recommendations for Future Research

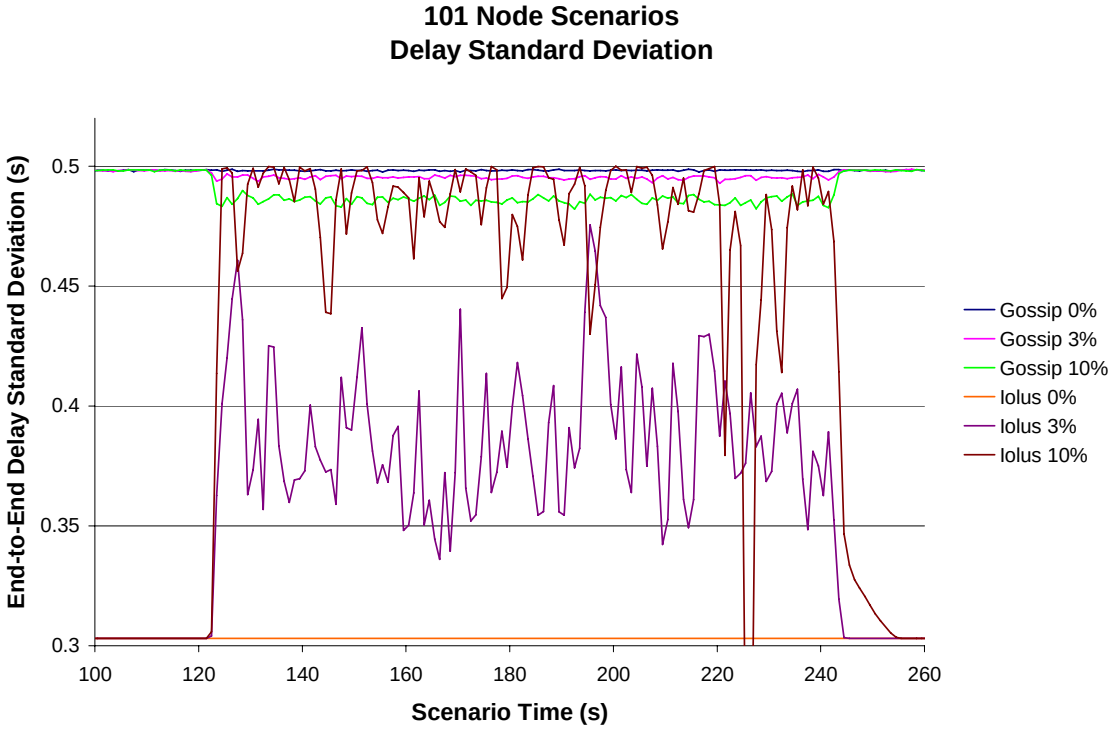
Wireless networks drive the need for further development of secure transmission protocols which are both scalable and fault-tolerant. This research considered a mesh topology. Further research could analyze the performance of various topologies.

A variety of established unicast security protocols could be incorporated into the key distribution scheme. This could be expanded to hybrid networks using a variety of protocols for each link in the system.

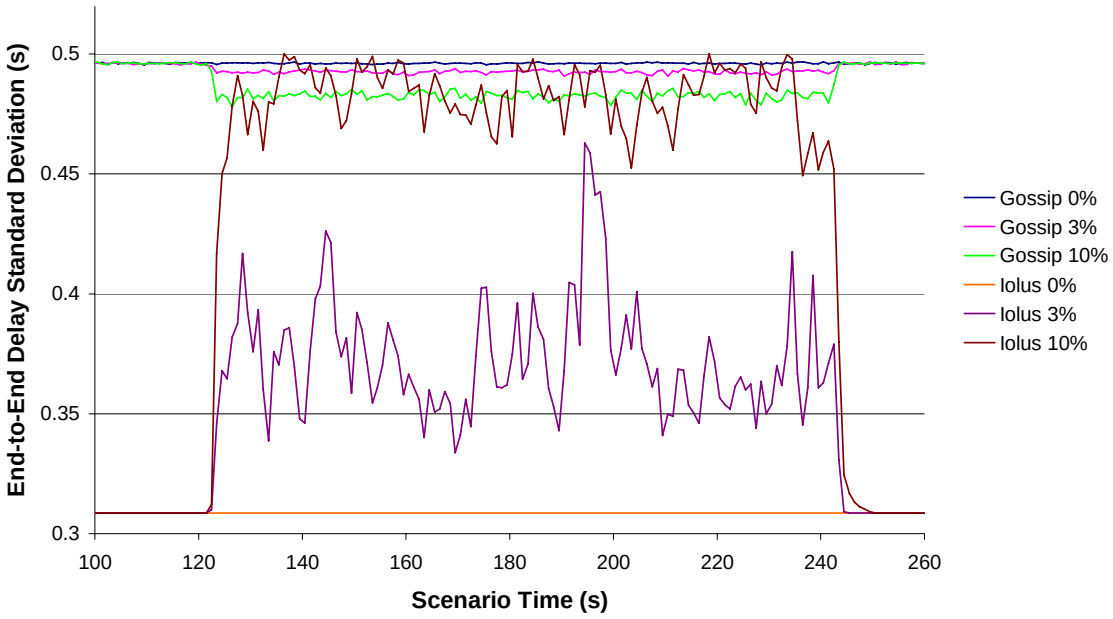
The experiments in this research can also be expanded to look at the effect of group membership changes. A rekey is forced by a join or a leave. In either case, the rekey is done by the sender changing the message key and updating the key ID attached to new messages. In some cases, this rekey may not be necessary. For example, it may be inconsequential whether a node receives the remaining part of a message when they have already received the first part. This can be left to the application.

Appendix A: Additional Charts

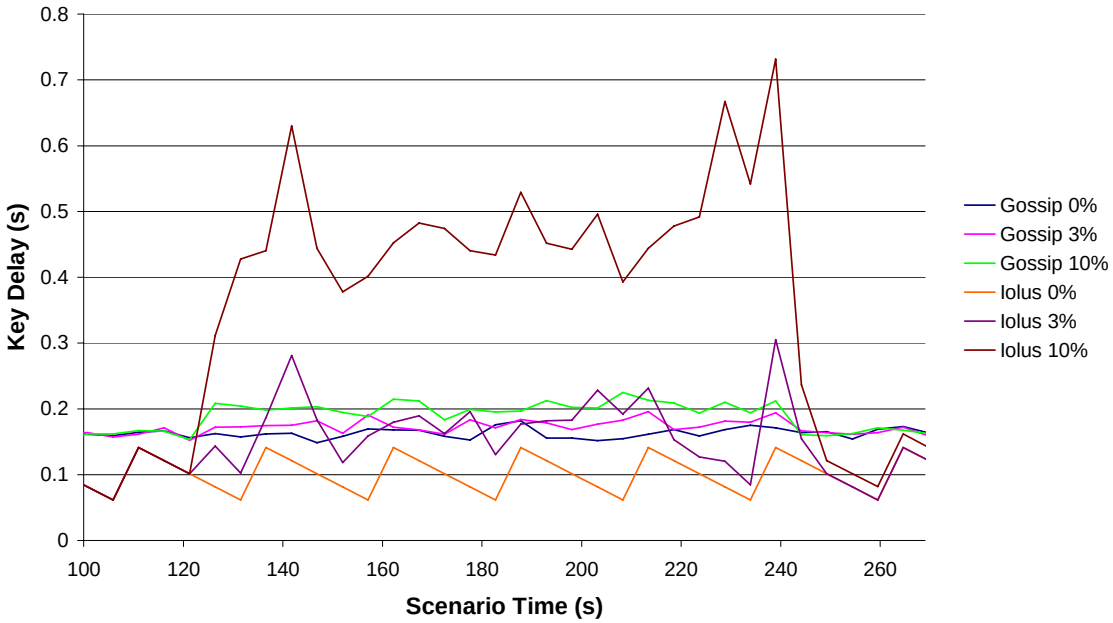


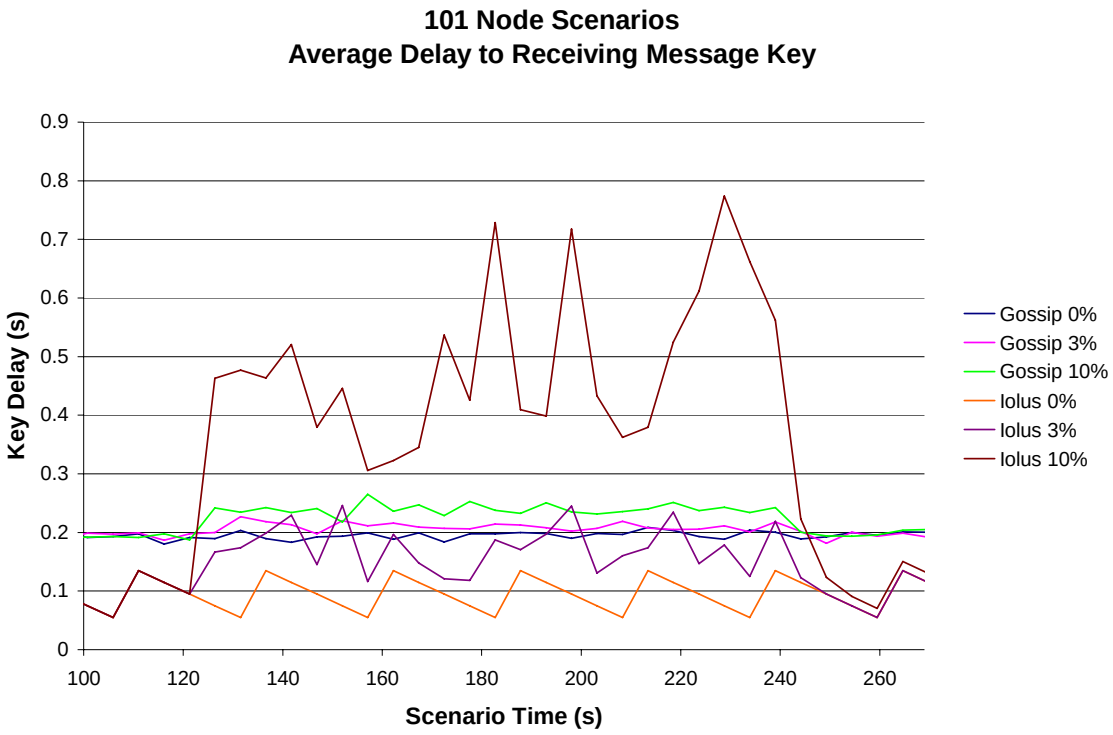


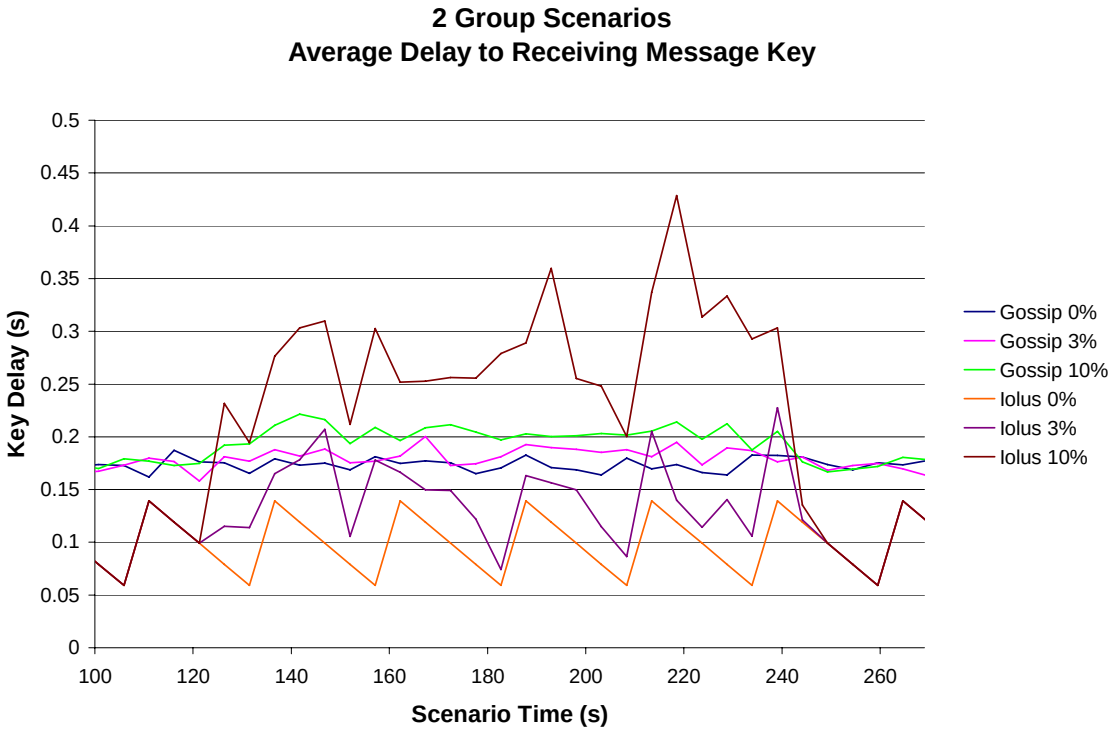
2 Group Scenarios
Delay Standard Deviation

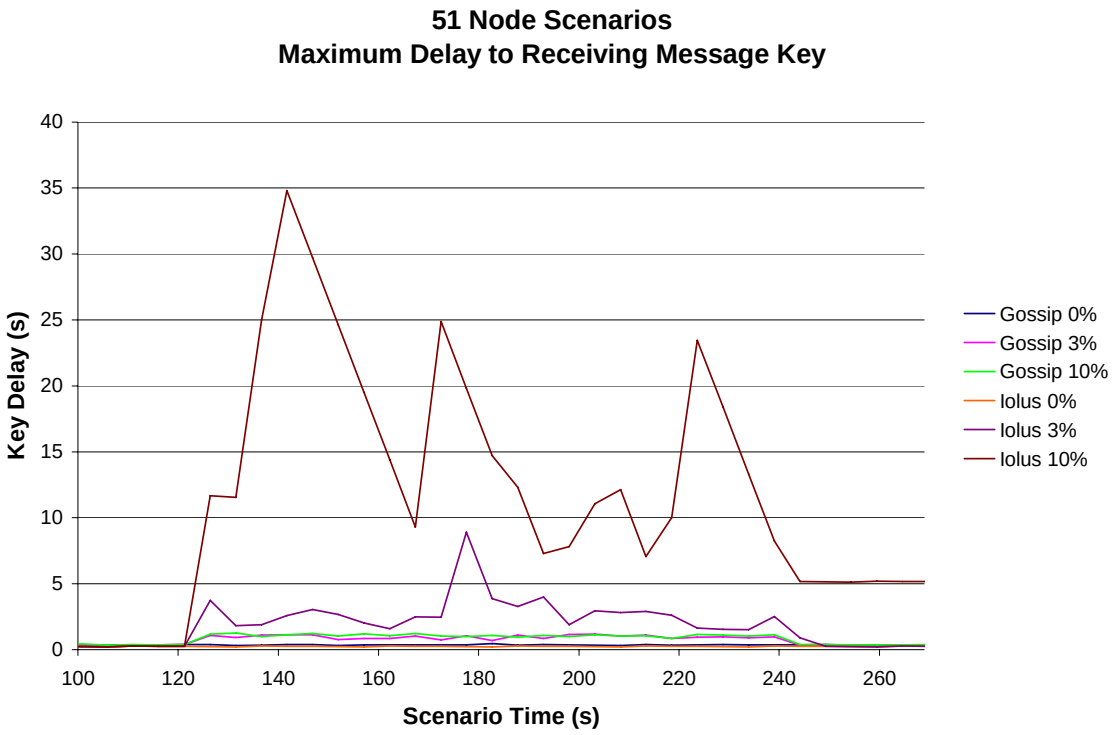


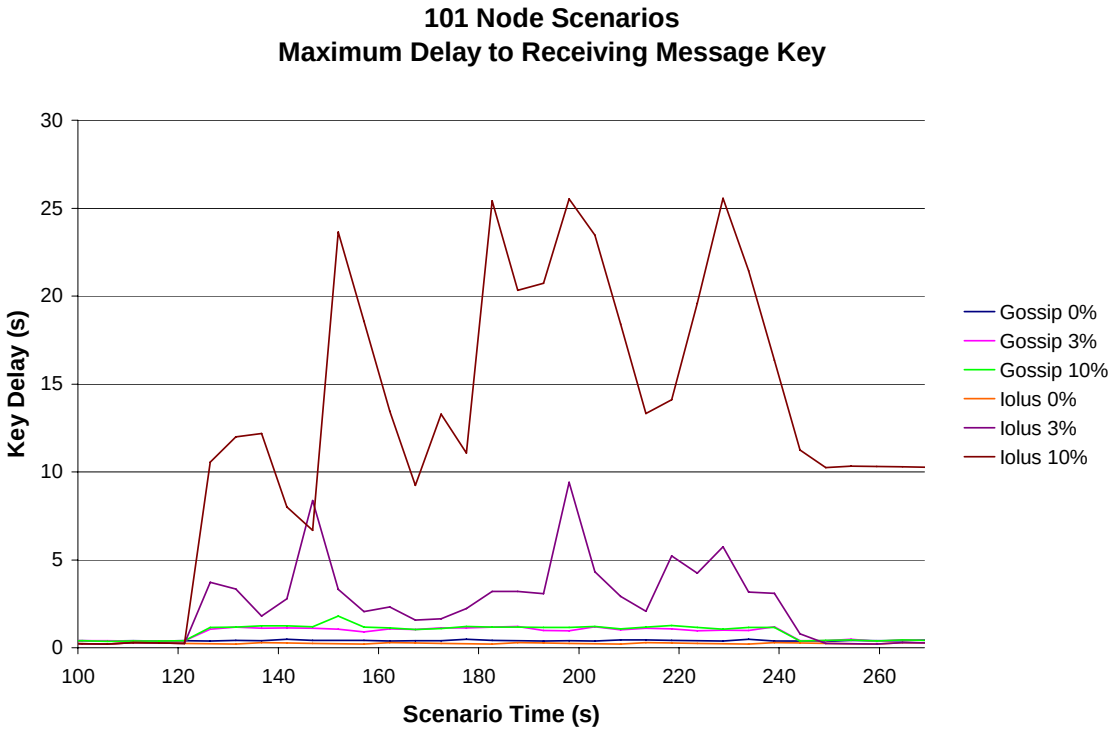
51 Node Scenarios
Average Delay to Receiving Message Key

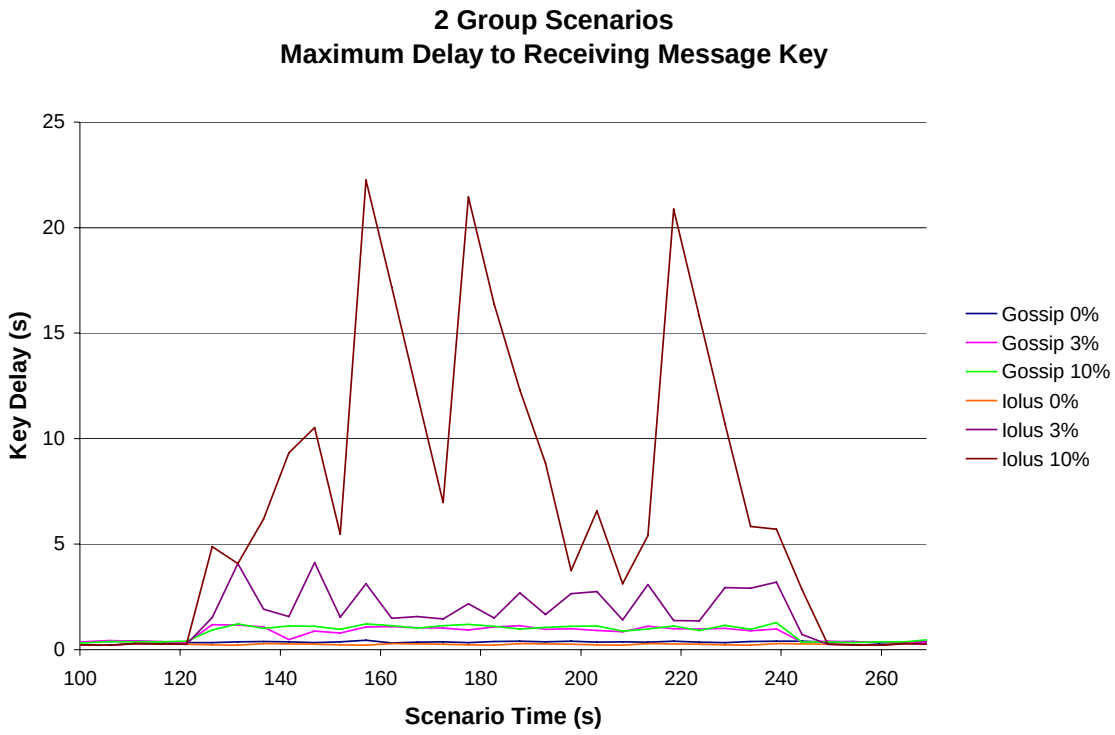


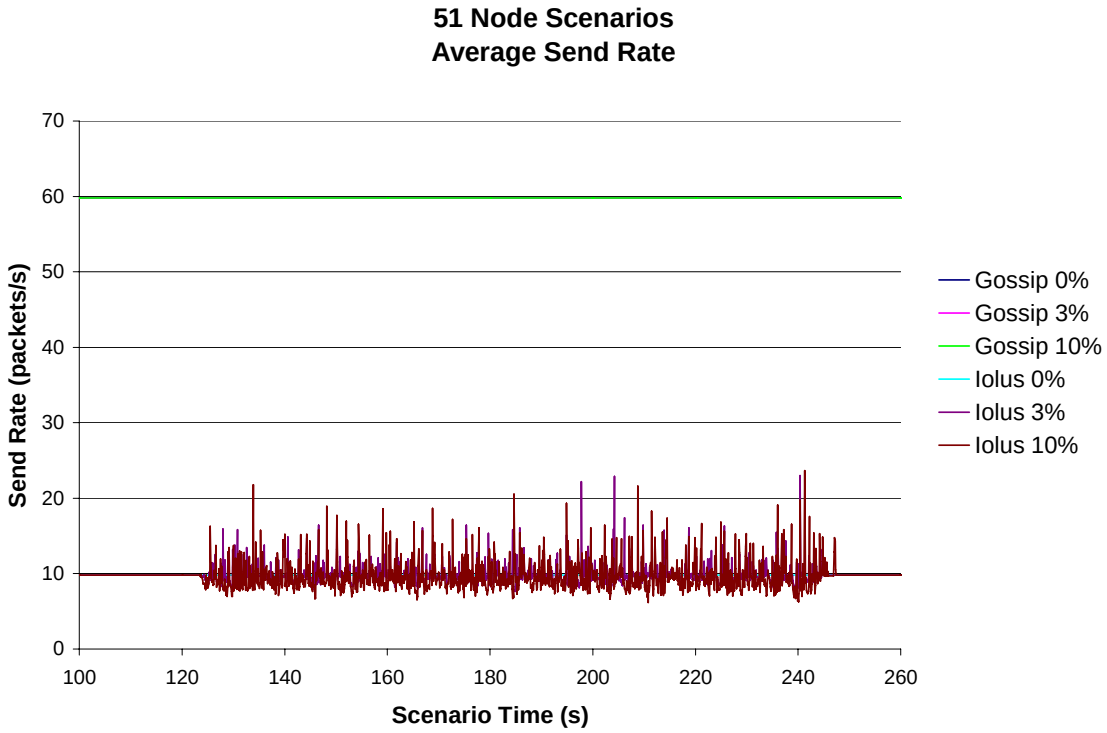

101 Node Scenarios
Average Delay to Receiving Message Key
Key Delay (s)
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

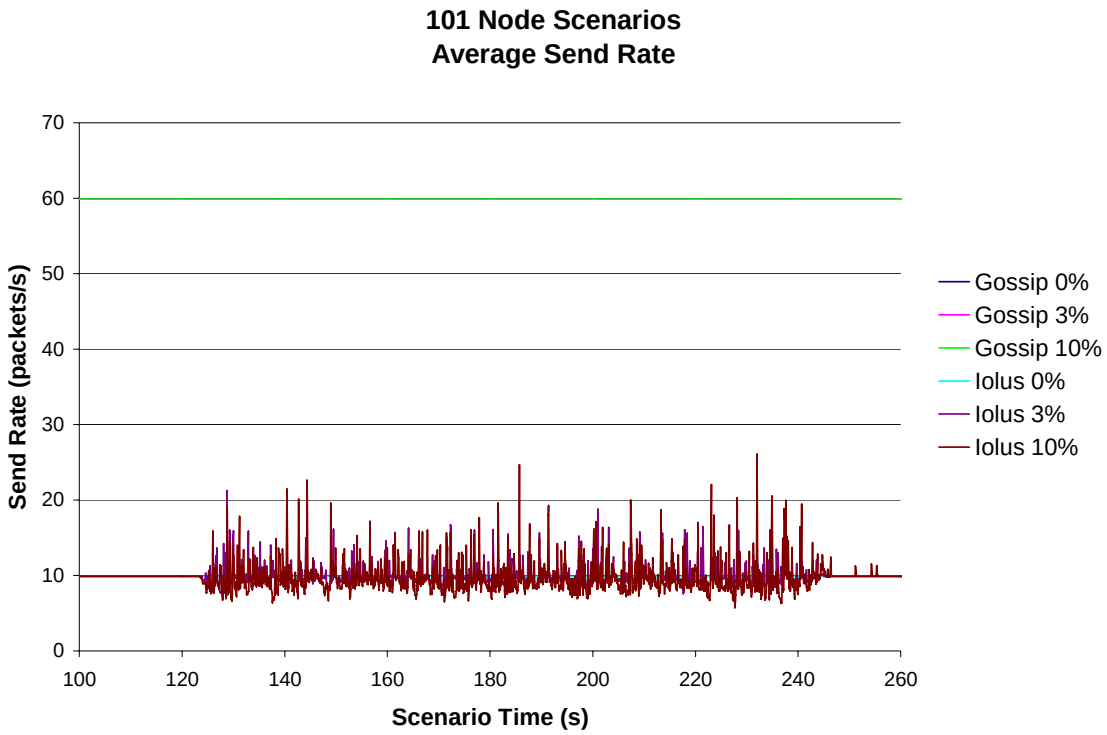

2 Group Scenarios
Average Delay to Receiving Message Key
Key Delay (s)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

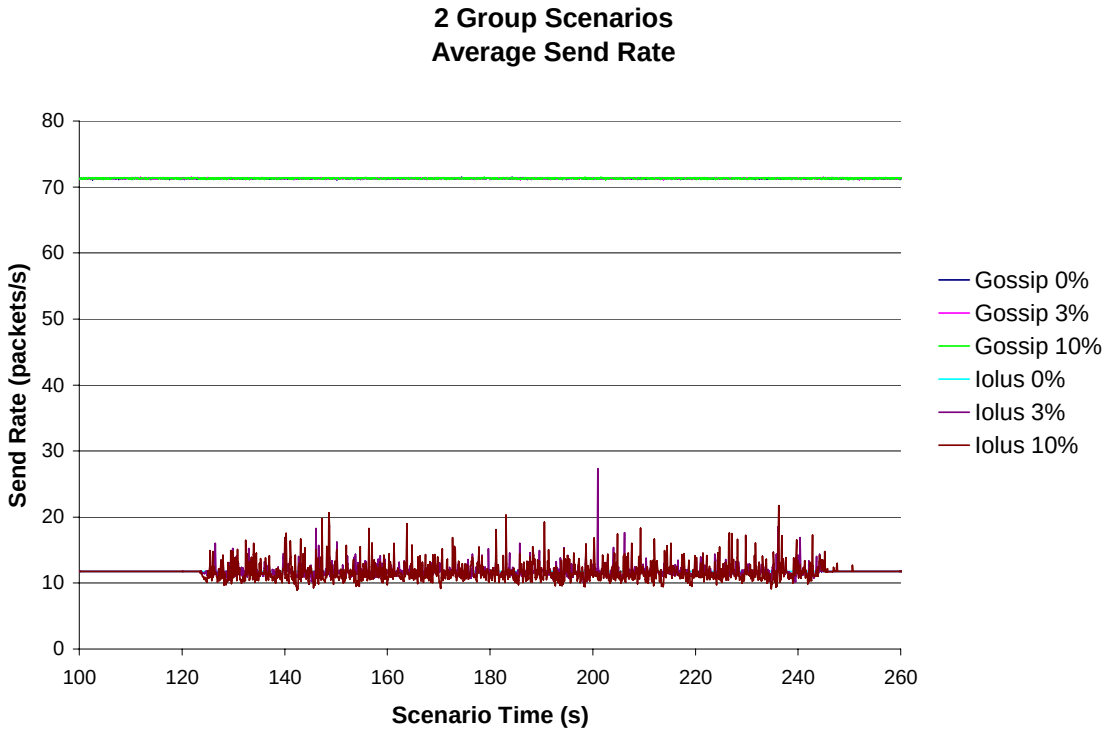

51 Node Scenarios
Maximum Delay to Receiving Message Key
Key Delay (s)
Scenario Time (s)
0
5
10
15
20
25
30
35
40
100
120
140
160
180
200
220
240
260
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

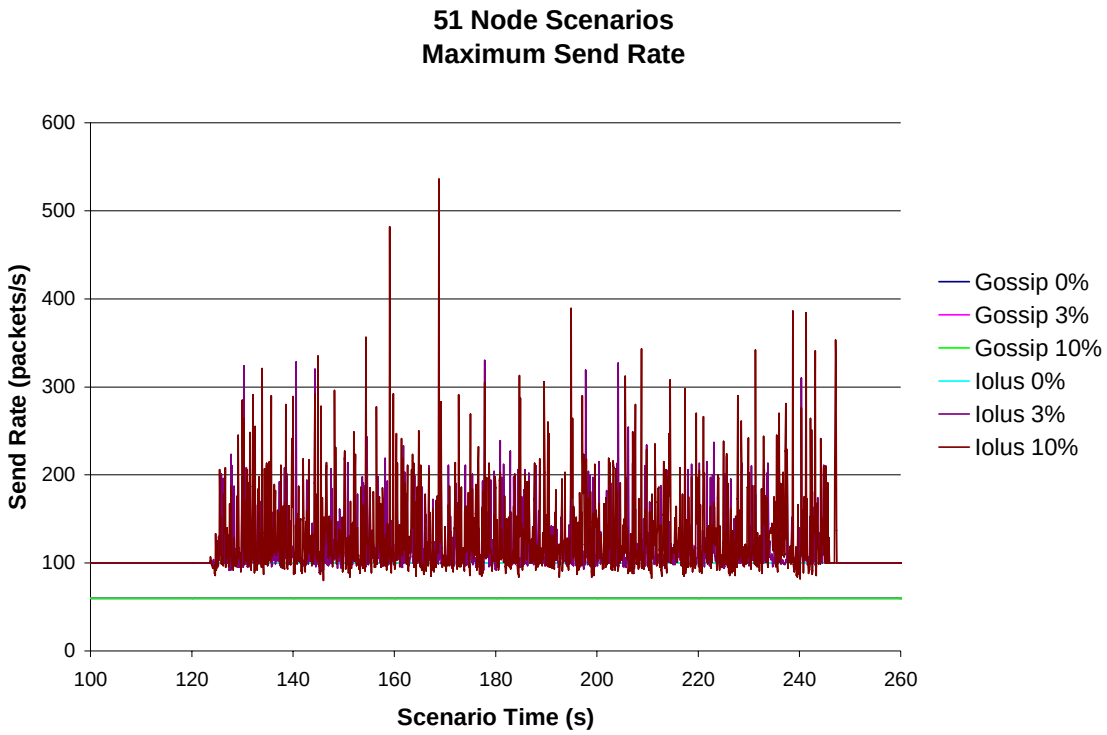

101 Node Scenarios
Maximum Delay to Receiving Message Key
Key Delay (s)
Scenario Time (s)
0
5
10
15
20
25
30
100
120
140
160
180
200
220
240
260
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

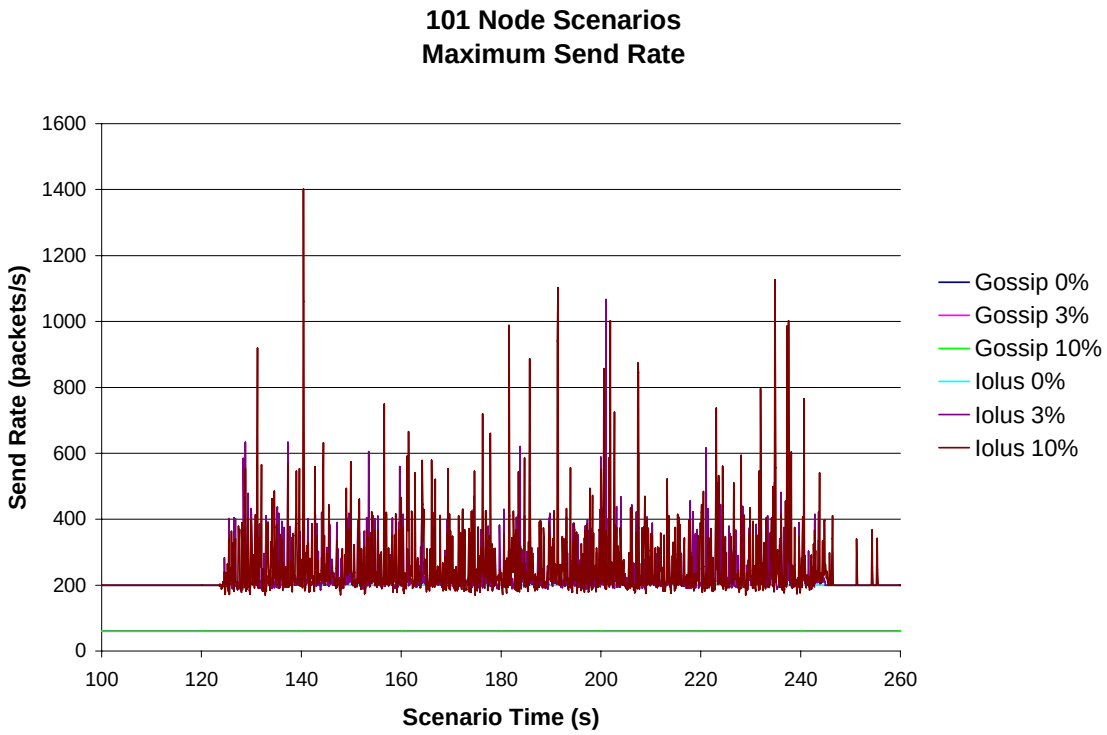

2 Group Scenarios
Maximum Delay to Receiving Message Key
Key Delay (s)
25
20
15
10
5
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

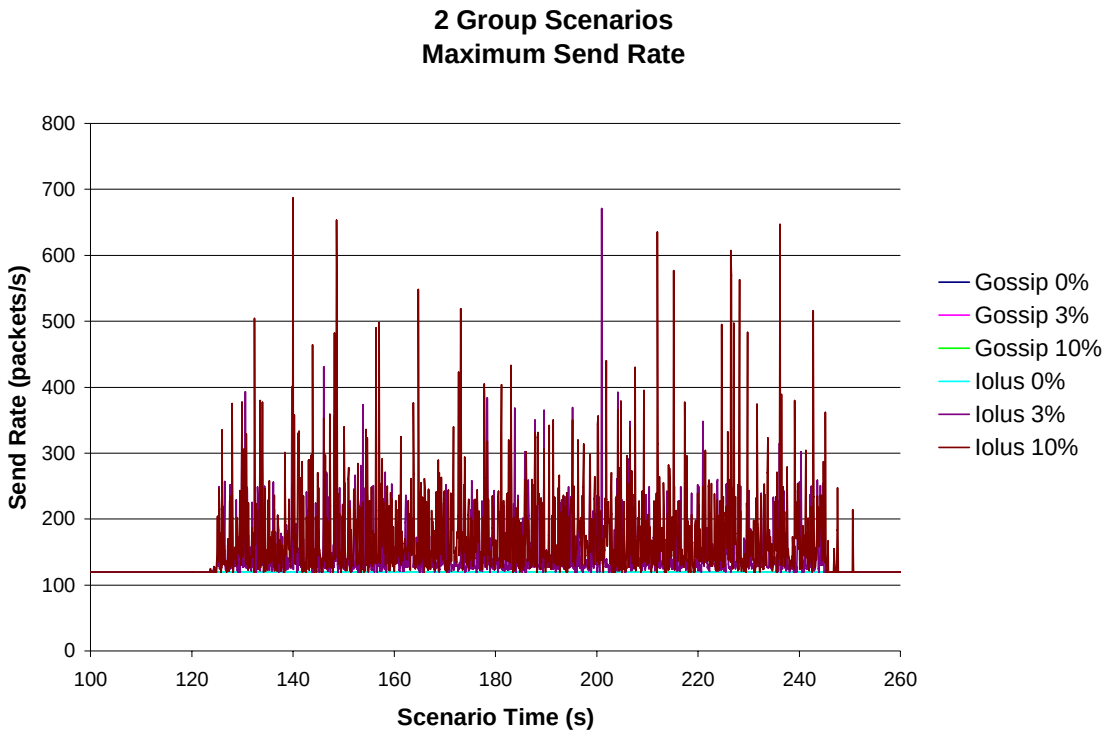

51 Node Scenarios
Average Send Rate
Send Rate (packets/s)
70
60
50
40
30
20
10
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

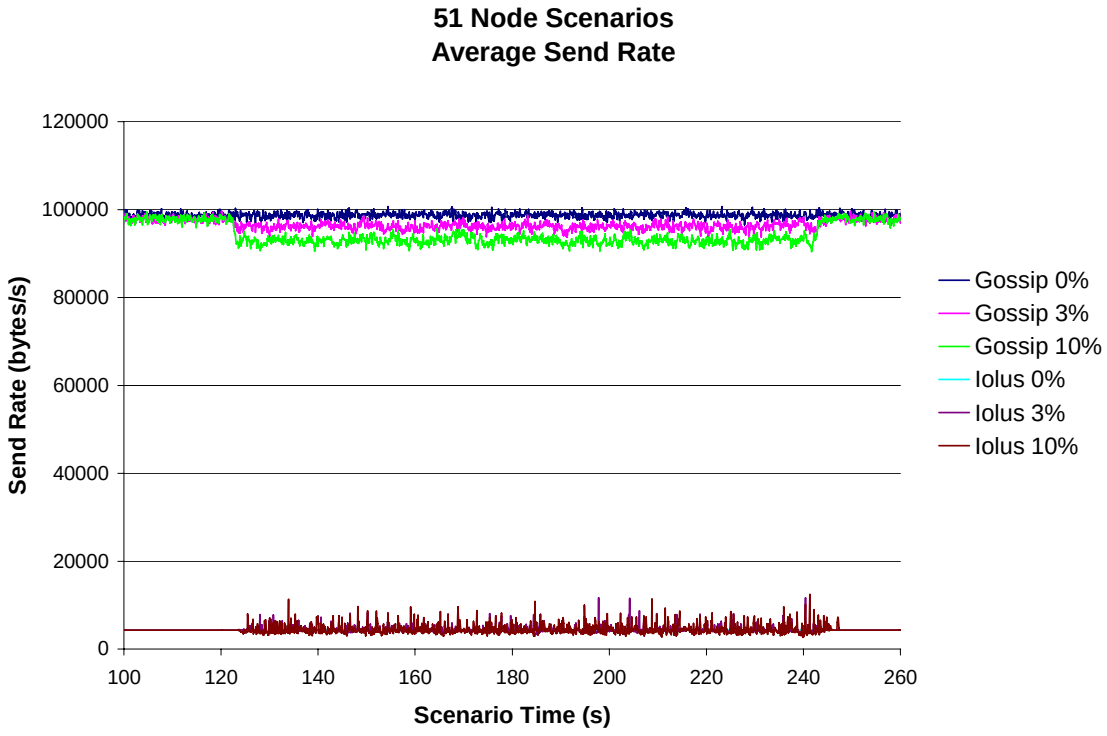

101 Node Scenarios
Average Send Rate
Send Rate (packets/s)
70
60
50
40
30
20
10
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

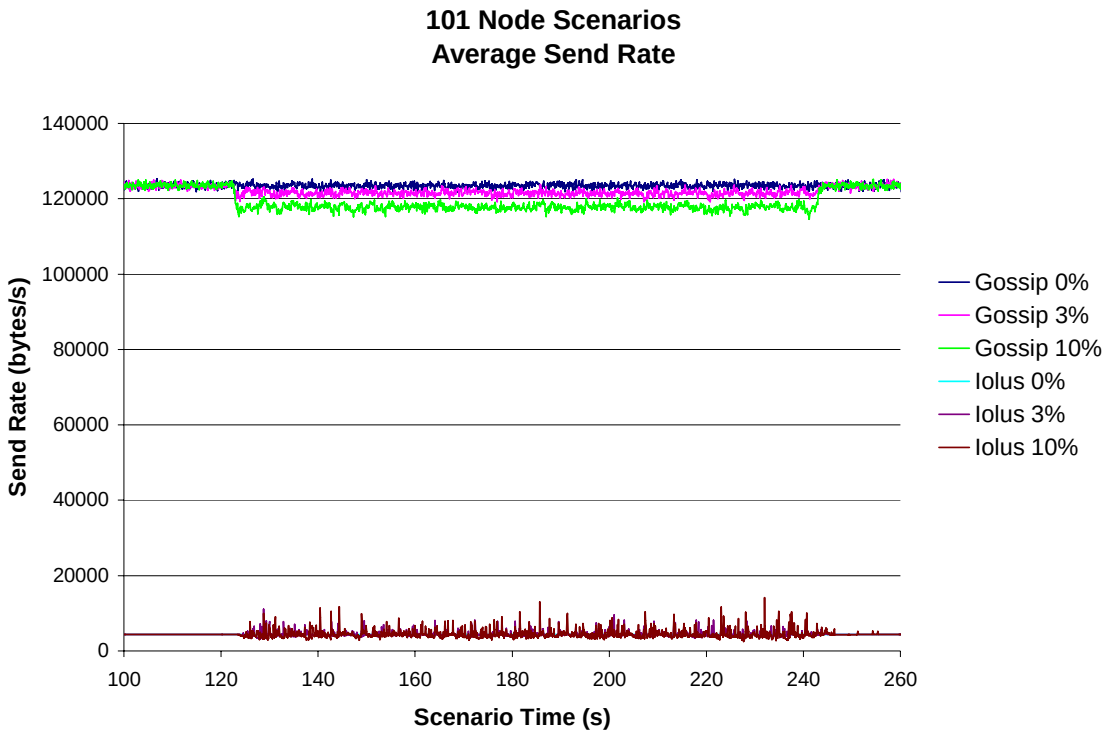

2 Group Scenarios
Average Send Rate
Send Rate (packets/s)
80
70
60
50
40
30
20
10
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

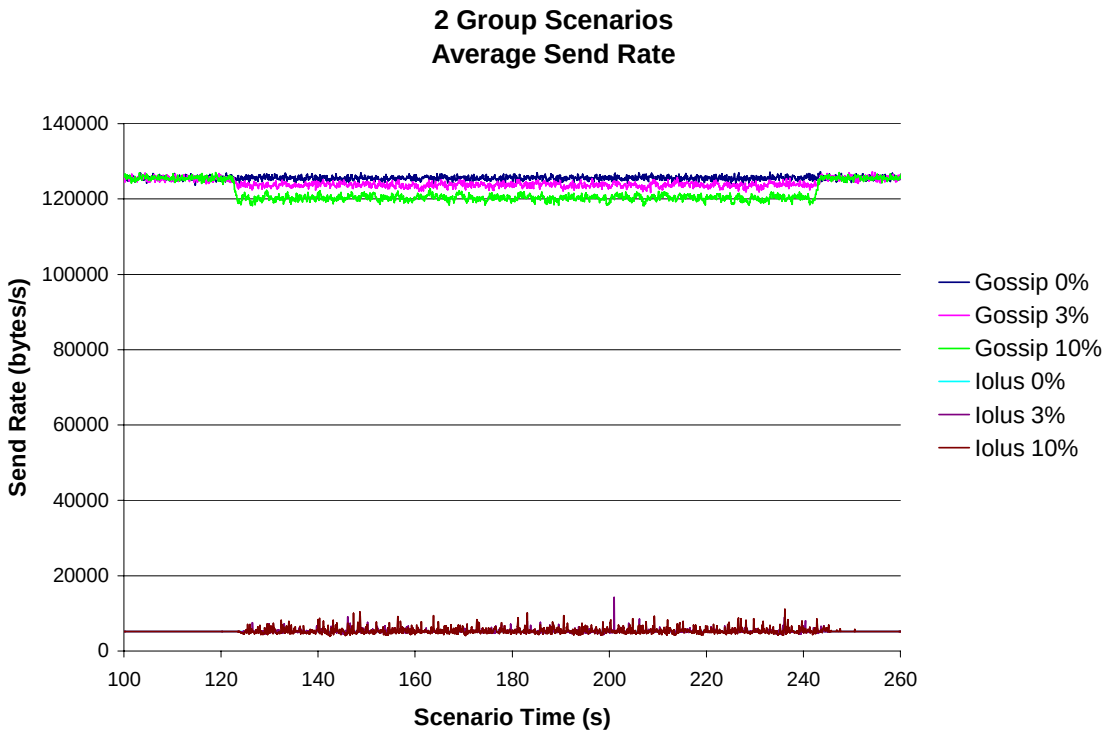

51 Node Scenarios
Maximum Send Rate
Send Rate (packets/s)
600
500
400
300
200
100
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

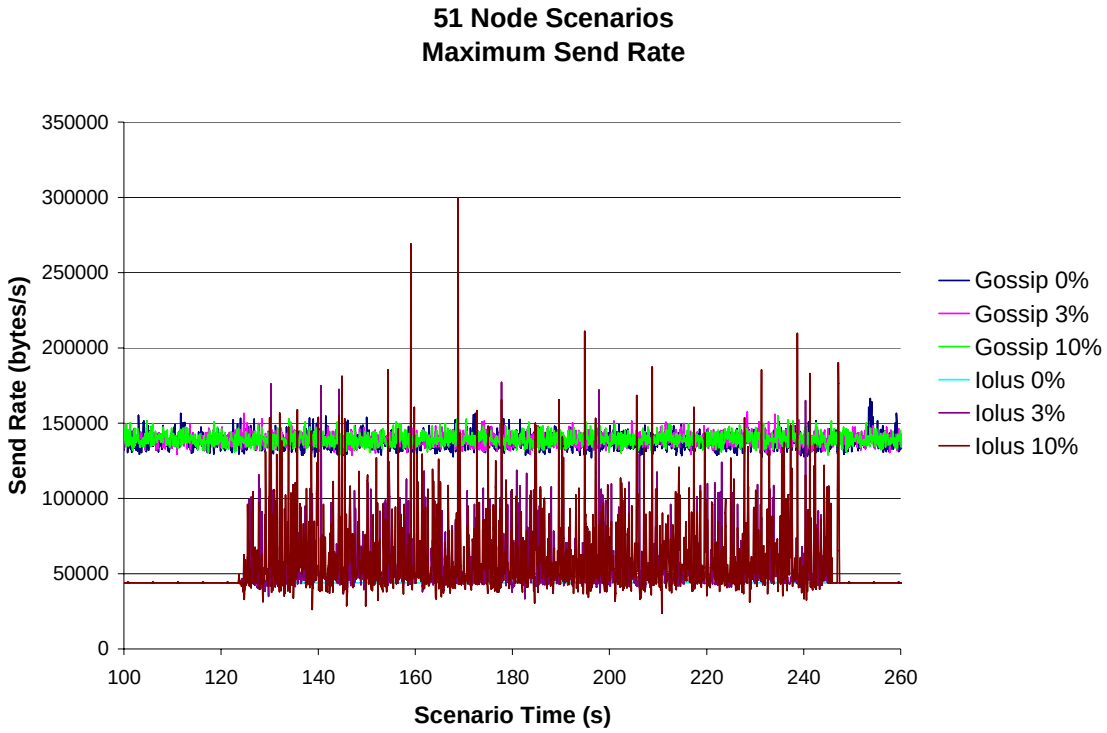

101 Node Scenarios
Maximum Send Rate
Send Rate (packets/s)
1600
1400
1200
1000
800
600
400
200
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

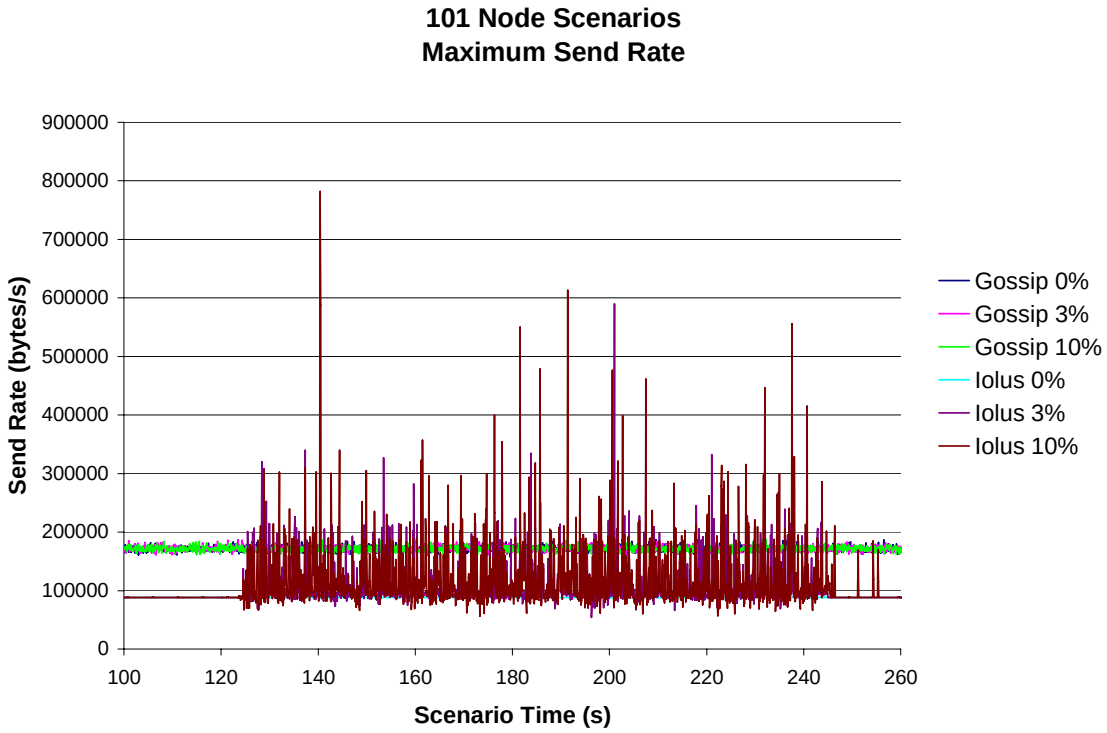

2 Group Scenarios
Maximum Send Rate
Send Rate (packets/s)
800
700
600
500
400
300
200
100
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

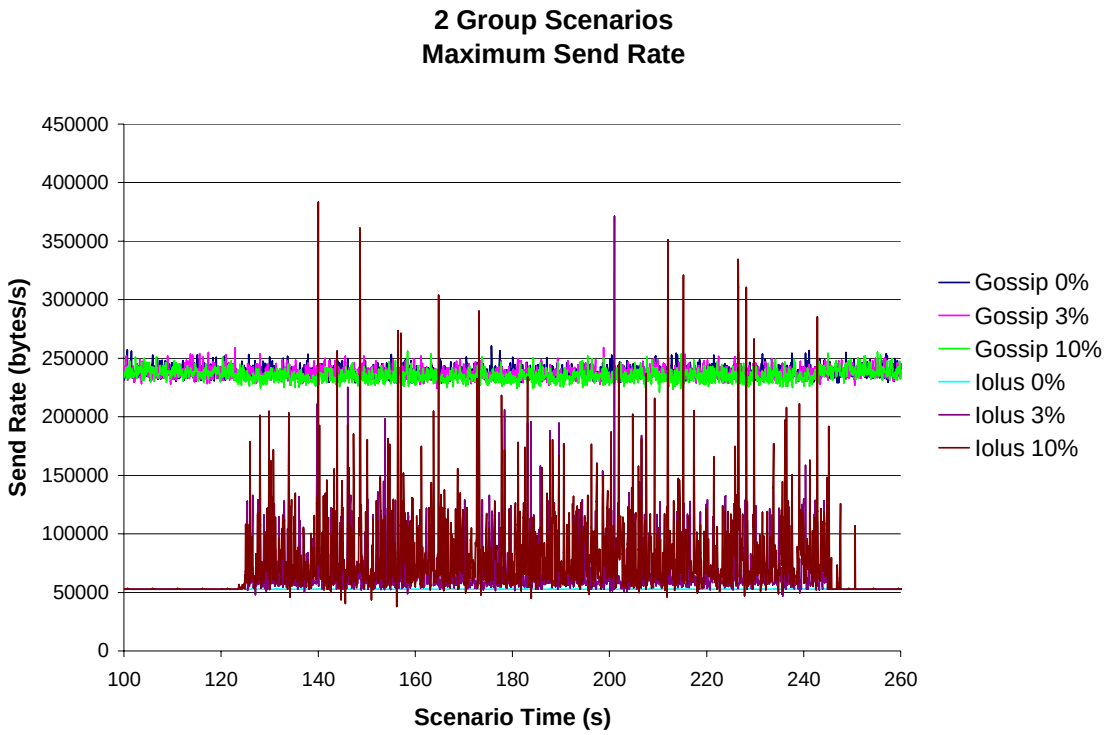

51 Node Scenarios
Average Send Rate
Send Rate (bytes/s)
120000
100000
80000
60000
40000
20000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%


101 Node Scenarios
Average Send Rate
Send Rate (bytes/s)
140000
120000
100000
80000
60000
40000
20000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%


2 Group Scenarios
Average Send Rate
Send Rate (bytes/s)
140000
120000
100000
80000
60000
40000
20000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%


51 Node Scenarios
Maximum Send Rate
Send Rate (bytes/s)
350000
300000
250000
200000
150000
100000
50000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%


101 Node Scenarios
Maximum Send Rate
Send Rate (bytes/s)
900000
800000
700000
600000
500000
400000
300000
200000
100000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%


2 Group Scenarios
Maximum Send Rate
Send Rate (bytes/s)
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
100
120
140
160
180
200
220
240
260
Scenario Time (s)
Gossip 0%
Gossip 3%
Gossip 10%
Iolus 0%
Iolus 3%
Iolus 10%

Bibliography

Birman, Kenneth, Mark Hayden, Oznur Ozkasap, Zhen Xiao, Mihai Budiu, and Yaron Minsky. "Bimodal Multicast," *ACM Transactions on Computer Systems,* Volume 17. Number 2. 41-88, May 1999.

Breslau, Lee, Deborah Estrin, Kevin Fall, Sally Floyd, John Heidemann, Ahmed Helmy, Polly Huang, Steven McCanne, Kannan Varadhan, Ya Xu, and Haobo Yu. "Advances in Network Simulation," *IEEE Computer*, Volume 33. Number 5. 59-67, (May 2000).

Huang, Qiang, Johnas Cukier, Hisashi Kobayashi, Bede Liu, and Jinyun Zhang. "Fast Authenticated Key Establishment Protocols for Self-Organizing Sensor Networks," *WSNA '03*, 141-150. San Diego: ACM Press, 2003.

McDaniel, Patrick, Atul Prakash, and Peter Honeyman. "Antigone: A Flexible Framework for Secure Group Communication," *Proceedings of the Eighth USENIX Security Symposium*, 99-114, Berkeley: USENIX Assoc, 1999.

Mittra, Suvo. "Iolus: A Framework for Scalable Secure Messaging," *SIGCOMM '97*, 277-288. New York: ACM Press, 1997.

Molva, Refik and Alain Pannetrat. "Scalable Multicast Security with Dynamic Recipient Groups," *ACM Transactions on Information and System Security*, Volume 3. Number 3. 136-160, (August 2000).

Rivest, R. L. "The RC5 Encryption Algorithm," *Fast Software Encryption. Second International Workshop Proceedings*, 86-96, Berlin: Springer-Verlag, 1995.

Rivest, R.L., A. Shamir, and L. Aldeman. "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, Volume 21. Number 2. 120-126, (February 1978).

Rodeh, Ohad, Kenneth Birman, and Danny Dolev. "The Architecture and Performance of Security Protocols in the Ensemble Group Communication System: Using Diamonds to Guard the Castle," *ACM Transactions on Information and System Security*, Volume 4. Number 3. 289-319, (August 2001).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
22-03-2007	Master's Thesis	Sept 2004 – March 2007

4. TITLE AND SUBTITLE

On-Demand Key Distribution for Mobile Ad-Hoc Networks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Graham, Daniel F.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GCS/ENG/07-12

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Dr. David R. Luginbuhl
Air Force Office of Scientific Research
875 N. Randolph Street
Arlington, VA 22203-1768
david.luginbuhl@afosr.af.mil (703) 696-6207

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Mobile ad-hoc networks offer dynamic portable communication with little or no infrastructure. While this has many benefits, there are additional shortcomings specific to wireless communication that must be addressed. This research proposes gossip-based on-demand key distribution as a means to provide data encryption for mobile ad-hoc networks. This technique uses message keys to avoid encrypting and decrypting a message at every node. Other optimizations used include secure channel caching and joint rekey messages. The use of gossip makes the scheme robust to node failure. Experimental results show only a 15% increase in end-to-end delay with a node failure rate of 10%. The percentage of messages successfully delivered to nodes stays between 91-98% under the same 10% node failure rate. The network load is distributed to all nodes in the group preventing overload and single points of failure.

15. SUBJECT TERMS

Data Transmission Security, Mobile, Computer Networks, Wireless Communications

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Kenneth Hopkinson (ENG)
U	U	U	UU	72	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636, ext 4579 (Kenneth.Hopkinson@afit.edu)